НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
ФАКУЛЬТЕТ ІНФОРМАТИКИ ТА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ
Кафедра автоматизованих систем обробки інформації і управління

УДК: 004.912

«До захисту допущено»
В.о. завідувача кафедри
_____ *О.А.Павлов*
(підпис) (ініціали, прізвище)
“___” _____2019 р.

Дипломний проект
на здобуття ступеня бакалавра

з напряму підготовки *6.050101 «Комп’ютерні науки»*

на тему: «*Інформаційна система побудови індивідуальних освітніх траєкторій*»

Виконав: студентка *4* курсу, групи *ІС-52*
Дворник Вікторія Анатоліївна _____
(прізвище, ім’я, по батькові) (підпис)

Керівник *доц., к.т.н., доц. Ковалюк Т.В.* _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант з графічної документації *старший викладач Халус О.А.* _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Рецензент *доц., к.т.н., доц. Пасько В.П.* _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Засвідчую, що у цьому дипломному проекті немає запозичень з праць інших авторів без відповідних посилань.
Студент Дворник В.А. _____
(підпис)

Київ – 2019 р.

АНОТАЦІЯ

Структура та обсяг роботи. Пояснювальна записка дипломного проекту складається з п’яти розділів, містить 109 сторінок, 22 рисунки, 45 таблиць, 1 додаток, 26 джерел.

Дипломний проект присвячений розробці системи побудови індивідуальних освітніх траєкторій для студентів вищих навчальних закладів, в яких враховуються психологічні, професійні та особисті якості студентів.

В дипломному проекті були розглянуті алгоритми та методи аналізу тексту: семантичний аналіз (метод латентно-семантичного аналізу) та визначення тематики тексту (метод латентного розміщення Діріхле).

У розділі з інформаційного забезпечення були визначені вхідні та вихідні дані до системи, була розроблена структура бази даних для збереження даних, яка відповідає поставленим цілям проекту.

У розділі з математичного забезпечення було обґрунтовано методи аналізу тексту. Був запропонований алгоритм визначення мотивації та визначення ключових слів областей знань.

У розділі з програмного забезпечення описані основні засоби розробки системи, висунуті вимоги до технічного забезпечення, обрано та обґрунтовано архітектуру програмного забезпечення.

У технологічному розділі описана інструкція користувача та проведене тестування системи.

ІНДИВІДУАЛЬНА ОСВІТНЯ ТРАЄКТОРІЯ, ПРЕТЕНДЕНТ, ТЬЮТОР, СТЕЙКХОЛДЕР, МОТИВАЦІЯ, КОМПЕТЕНТНІСТЬ, ОБЛАСТЬ ЗНАНЬ.

ABSTRACT

Structure and scope of work. Explanatory note of the diploma project consists of five sections, containing 109 pages, 22 figures, 45 tables, 1 annex, 26 sources.

The diploma project is devoted to the development of a system for constructing individual educational trajectories for students of higher educational institutions, which take into account the psychological, professional and personal qualities of students.

In the diploma project considered algorithms and methods of text analysis: semantic analysis (method of latent semantic analysis) and the topic definition of the text (the method of Latent Dirichlet allocation).

In the section on information support defined input and output data of the system, a database structure for data storage was developed that corresponds to the objectives of the project.

In the section on mathematical support, the methods of text analysis were substantiated. Was proposed he algorithm for determining the motivation and definition of the key words of the knowledge areas.

The software section describes the main tools of the system development, also describes the requirements for technical support, the software architecture was chosen and justified.

The technology section describes the user's manual and tests for a system.

INDIVIDUAL EDUCATION TRAJECTORY, APPLICANT, TUTOR, STEAKHOLDER, MOTIVATION, COMPETENCE, KNOWLEDGE AREA.

ЗМІСТ

ВСТУП

Протягом усього життя людина розвивається, отримує нові знання та пізнає щось нове. Для подальшого прогресу людині необхідно постійно вдосконалюватись. Освіта завжди була і буде ключем до гармонічного розвитку особистості. Навчаючись, людина розширює кругозір, отримує новий досвід, а також поповнює свої знання та вміння.

Освіта є основою духовного, фізичного, інтелектуального та культурного розвитку особистості, її успішної соціалізації, а також економічного добробуту. Якісна освіта – це запорука розвитку держави та суспільства, об'єднаного спільними цінностями та культурою. Мета освіти – усебічний розвиток людини як особистості та найвищої цінності демократичного суспільства, її інтелектуальних, творчих і фізичних здібностей, а також її талантів, формування цінностей і необхідних для успішної самореалізації компетентностей.

Освітній процес – система науково-методичних та педагогічних заходів, котрі спрямовані на розвиток людини шляхом формування і застосування її компетентностей. Компетентність – це динамічна комбінація знань, умінь, навичок, поглядів, цінностей, способів мислення та інших особистісних якостей, яка визначає здатність особи реалізовуватись у суспільстві, вести професійну та подальшу навчальну діяльність [1].

На сьогоднішній день все популярнішою стає модель індивідуального підходу в навчанні, яка спрямований на підтримку ефективності самого процесу навчання. Індивідуалізація навчання відбувається шляхом реалізації індивідуальних освітніх траєкторій.

Індивідуальна освітня траєкторія – це персональний шлях реалізації потенціалу претендента на здобування освіти, котрий формується з урахуванням його інтересів, здібностей, потреб, мотивації, а також можливостей та досвіду, грунтується на виборі претендента на освіту форм,

видів та темпу здобуття освіти, суб'єктів навчальної діяльності та запропонованих ними програм освіти, навчальних дисциплін, а також рівня їх складності, методів та засобів навчання. Індивідуальна освітня траєкторія в освітньому закладі може бути реалізована у вигляді індивідуального навчального плану.

Індивідуальний навчальний план – це документ, котрий визначає послідовність, форму та темп засвоєння претендентом на здобуття освіти освітніх компонентів навчальної програми із метою реалізації його особистої індивідуальної освітньої траєкторії (шляху). Він розробляється освітнім закладом у взаємодії зі здобувачем освіти за наявності потрібних для цього ресурсів [1].

Для формування індивідуальної освітньої траєкторії необхідні такі суб'єкти:

– претендент – особа, яка прагне здобувати освіту за формою індивідуальних освітніх траєкторій;

– тьютор – суб'єкт освіти та педагогічної діяльності, персональний наставник претендента, допомагає у процесі визначення дисциплін;

– стейкхолдер – зацікавлена особа, представник ІТ-компанії, який бере на себе відповідальність визначення якості освіти, яку отримуватиме претендент, та її відповідність певній професії.

Дипломний проект присвячений розробці системи побудови індивідуальних освітніх траєкторій.

Практичне значення одержаних результатів. Розроблено алгоритм для визначення мотивації та алгоритм визначення ключових слів областей знань.

Публікації. Результати роботи були опубліковані у тезах доповіді у науковому збірнику «ІНТЕЛЕКТУАЛЬНІ РІШЕННЯ. Обчислювальний інтелект. Матеріали V Міжнародної науково-практичної конференції»

Міжнародного наукового симпозіуму «Інтелектуальні рішення» (ст. 193-194) та тезах доповіді у науковому збірнику «Секція кафедри автоматизованих систем обробки інформації і управління. Метеріали конференції» ІІ Всеукріїнської науково-практичної конференції молодих вчених та студентів «Інформаційні системи та технології управління – ІСТУ-2019» (ст. 109-113).

1 ЗАГАЛЬНІ ПОЛОЖЕННЯ

1.1 Опис предметного середовища

Отримання якісної освіти – важлива річ для сучасного студента. Самі студенти вважають, що освіта в університеті повинна бути націлена на ефективність, функціональність та індивідуальність. У сучасному освітньому процесі саме поняттю індивідуальності приділяється найменше уваги, хоча студент – це унікальна особистість зі своїми інтелектуальними спроможностями, інтересами та соціальним досвідом.

Загалом, методи навчання та його традиційні форми приводять обдарованих особистостей до загального, єдиного для всіх та стандартного освітнього шляху. Такий шлях спрямований на пасивне засвоєння важливих та менш важливих знань та вмінь, вимагають від студента посидючості, пильності та сумлінності, не розвиваючи в ньому прагнення до самореалізації і завзятості [2].

З роками форма здобуття знань видозмінювалась, і існує тенденція переходу від стандартної форми до сучасної моделі самонавчання. На сьогоднішній день люди прагнуть опановувати велику кількість знань та здобувати певні компетентності самостійно, цьому спонукає потреба приналежності до деякої соціальної групи, потреба до самовираження та зацікавленість власним благополуччям та можливостями.

Проте, інколи люди губляться в масі доступності знань. Саме тому потрібно активізувати становище системи освіти процесами, які будуть направлені на пошук більш ефективних форм освітньої діяльності, на створення необхідних умов навчання та розвитку, що допомогло б максимально розкрити здібності всіх студентів. Зважаючи на це, для формування низки компетентностей кожного студента для подальшого працевлаштування, максимально враховуючи індивідуальні особливості, в

освіті ідеальним сценарієм можна вважати індивідуалізацію освіти [2]. Це допоможе студенту досягти цілі, якої він прагне.

Такий персональний підхід можна забезпечити лише з використанням індивідуальних освітніх траєкторій, які розробляються для кожного конкретного студента, враховують його зону теперішнього та майбутнього розвитку.

Індивідуальна (персональна) освітня траєкторія – це організована програма дій студента на деякому визначеному термінами етапі його навчання.

Використання індивідуальних траєкторій дає повну картину про особистісний розвиток студента, дозволяє певною мірою з’ясувати причини його слабкої навчальної мотивації або, навпаки, побачити причини стійкої та позитивної мотивації [3].

Персоналізація навчання може бути розглянута як упорядкування навчального процесу, а також сукупність різних навчальних, педагогічних, психологічних, методичних і організаційно-управлінських заходів, що забезпечують індивідуальний підхід. При цьому вибір темпу навчання, способів, прийомів зумовлюється індивідуальними особливостями студентів. Введення індивідуальних траєкторій надає можливість створення таких умов, які здатні забезпечити активне стимулювання освітньої діяльності на основі самоосвіти, саморозвитку, самовираження в процесі оволодіння знаннями. Така система навчання дозволяє здійснити особистісно-орієнтований підхід в освітньому процесі, максимально враховує розумові здібності студентів [2].

1.1.1 Опис процесу діяльності

Типова послідовність дій для процесу побудови індивідуальної траєкторії представлена у графічному додатку. Діаграма діяльності містить такі доріжки:

– вхідна та вихідна інформація – опис вхідних та вихідних даних, яка використовується системою у ході бізнес-процесу;

– діяльність – стадії виконання бізнес-функцій;

– учасники процесу діяльності – актори, котрі займаються виконанням бізнес-функцій;

– підрозділ - відділ організаційної структури підприємства, котрому належить актор;

– бізнес-правила – опис правил та лімітів виконання бізнес-функцій процесу.

Процес формування індивідуальної навчальної траєкторії відбувається наступним чином: претендент проходить декілька видів тестів послідовно (психологічний тест, визначення лідерських якостей в команді, на профпридатність та професійні компетентності), а також виявляє власні побажання щодо індивідуальної траєкторії. Система аналізує результати: виділяє у студента психологічні якості, формує психологічний портрет, формує список професій, які підходять претенденту, визначає його знання, вміння, досвід, а також робить висновок щодо наявності мотивації. Претендент може переглянути всі результати.

Якщо мотивація чи компетентності у студента відсутні, то він переходить до типового навчального плану, вважається, що індивідуальна навчальна траєкторія йому не потрібна.

Після цього відбувається процес формування індивідуальної освітньої траєкторії: система виконує кореляцію професійних компетентностей з ІТ-професією, також визначає області знань для обраної користувачем професії. Далі проводиться визначення ключових слів із обраних областей знань. Ключові слова передаються тьюторові для формування списку дисциплін для претендента. Претендент отримує особистий список дисциплін і має можливість обрати лише 8 дисциплін на свій розсуд. Обрані дисципліни надсилаються стейкхолдеру для затвердження. Якщо дисципліни були

затверджені, вони формують навчальний план та відображаються в особистому кабінеті претендента.

1.1.2 Опис функціональної моделі

Опишемо функціональну модель системи за допомогою діаграми IDEF0.

Діаграма IDEF0 – це функціональна модель, яка є ядром побудови всіх інших конструкцій, вона пов’язує воєдино інформаційні та матеріальні потоки, оргструктуру, що управляють і самою діяльність компанії [4].

Схема структурна контекстної моделі системи представлена у графічному додатку.

Вхідними даними є: заявка на формування індивідуальної освітньої траєкторії.

Вихідні дані: індивідуальна освітня траєкторія.

Управління представлене такими нормативними документами: освітні стандарти, освітні програми, професійні стандарти.

Механізмами є претенденти, стейкхолдери, тьютори.

Побудувавши до системи формування індивідуальної навчальної траєкторії контекстну діаграму, декомпозуємо її на такі підсистеми (роботи):

– проходження тестів;

– визначення компетентностей;

– визначення професій;

– перевірка мотивації;

– формування індивідуальної освітньої траєкторії.

Схема структурна функціональної моделі IDEF0 представлена на у графічному додатку.

На цій схемі провізуалізовано інформаційні потоки між підсистемами, елементами управління ними та механізмами, які безпосередньо будуть з ними співпрацювати.

Також на схемі наочно видно, на якому етапі задіяні ті чи інші керуючі елементи і механізми.

При проходженні тестів використовується вхідна заявка на формування індивідуальної освітньої траєкторії. Система надає результати у вигляді психологічного портрету та особистісних якостей. Претендент визначає свої компетентності за допомогою вже відомих стандартів. Для визначення професії, яка найкраще підходить претендентові, потрібні вже відомі компетентності (знання, вміння, досвід). Після цього визначається мотивація студента: залежно від його побажань та обраної професії. Для формування індивідуальної навчальної траєкторії необхідно задіяти усі механізми бізнес-процесу та керуючі елементи.

1.2 Огляд наявних аналогів

У ході пошуку аналогічних за функціональністю систем було знайдено наступні системи:

– система «ТРАЕКТОРИЯ.ОНЛАЙН» [5];

– система Dean-Dashboard.com [6];

– система в Тюменському університеті [7].

Основними функціями системи «ТРАЕКТОРИЯ.ОНЛАЙН» є [5]:

– проведення тестувань з психології, профорієнтації: на основі підсумків тестів система підбирає місця навчання і роботи, де студент може себе реалізувати;

– складання психологічного портрету для глибшого пізнання талантів, здібностей і схильностей: це допоможе правильно вибрати місце навчання і роботи;

– знаходження секції для спорту та громадських організацій для улюблених занять: з урахуванням результатів тесту система складе список цікавих студенту організацій для дозвілля, спорту та волонтерства;

– тестування з профорієнтації та надання список відповідних професій, що відповідають талантам, здібностям і характеру студента;

– на основі підсумків тестів система підбере коледжі, вузи і освітні програми, які підходять для навчання;

– система складає список компаній, які підходять для роботи конкретному претенденту.

Розглядаючи систему Dean-Dashboard.com, можна виділити її основні функції [6]:

– дозволяє будувати особисту траєкторію без будь-якої допомоги, автори проекту не несуть відповідальності за навчання відповідно до сформованого шляху;

– можливість використання інструменту для створення та відстеження власного шляху навчання – він потрібний для особистого використання студентами онлайн-курсів;

– можливість використання інструменту для створення навчального плану для певного наукового ступеня.

Кожна траєкторія системи може містити:

– курси – будь-який курс з каталогу можна додати до шляху;

– групи альтернатив – це набір курсів з однаковою тематикою для вивчення.

– групи елективів – додається до шляху у випадку, коли студент може вибрати тему, яку він хотів би вивчити як частину свого шляху, використовується для «загального вивчення»;

– практичні тренінги;

– іспити.

В Тюменському університеті вже з вересня 2017 року студенти чотирьох інститутів (Інституту філології та журналістики, Інституту хімії, Інституту психології і педагогіки та Інституту історії і політичних наук) навчаються за

новою освітньою моделлю [7]. Завдяки впровадженню системи індивідуальних освітніх траєкторій у студентів 1-го курсу є можливість самостійно формувати свій власний навчальний план за рахунок вибору навчальних курсів (елективів) з п’яти областей знань поряд з вивченням обов’язкових дисциплін. Цими областями знань є:

- природничі науки і технології;
- мистецтво;
- математика та інформатика;
- соціальна комунікація;
- науки про суспільство і людину.

Згідно з нормативними документами, вибір і вивчення елективних дисциплін стане можливим тільки з 2-го семестру. Поки ж, в 1-му семестрі, важливо «ввести» студентів в особливий режим вузівського навчального процесу і закласти базу, на основі якої вони зможуть надалі самостійно вибудовувати свій розклад. Відповідно до навчальних планів в 1-му семестрі студенти будуть вивчати тільки обов’язкові базові дисципліни. Обов’язковими для вивчення у вузі є: дисципліни професійного блоку (Major), які забезпечують формування професійних компетенцій і дисципліни загальноосвітнього блоку (Core), що забезпечують формування компетентностей учнів. Важливо, що обов’язкові дисципліни блоку Core єдині для студентів усіх напрямів підготовки, які беруть участь в експерименті. Це означає, що і філологи, і хіміки, і психологи, і історики, і ін. протягом 1-го семестру отримають однаковий обсяг базових загальноосвітніх знань.

Після вибору студентом, курси включаються в його індивідуальний навчальний план і стають обов’язковими для вивчення. Вибір елективних дисциплін здійснюється студентами в особистому кабінеті. У загальноосвітньому блоці в 2-му семестрі студенту належить вибирати 6 дисциплін з 5 блоків. На другому і наступному курсах – по 3 дисципліни з

будь-якої галузі знань. Елективна дисципліна буде реалізовуватися, якщо на неї записалося не менше 25 осіб. Якщо кількість бажаючих вивчати дисципліну буде занадто велика, то пріоритет при формуванні груп отримають студенти з більш високими балами.

Отже, вибираючи дисципліни з загальноосвітнього або професійного блоків відповідно до своїх інтересів, студент має можливість не тільки більш глибоко зануритися в освоєння дисциплін професійної сфери, але і отримає додаткові компетенції в інших областях. Це істотно підвищує шанси студента на ринку праці.

А. В. Проворова [8] стверджує, що при проектуванні освітньої програми учня навчальний процес піддається особистісній детермінації, суть якої виражається в узгодженні проблем учня з проблемами освітнього процесу при адекватній зміні останнього.

Під індивідуальною освітньою програмою розуміється програма розвитку учня, заснована на знанні його особливостей, як особистості. На відміну від навчальної програми, вона гнучко пристосована до можливостей учня, динаміки його розвитку під впливом навчання. Специфіка побудови індивідуального освітнього маршруту в МУК проявляється у великих можливостях вибору профільних занять (елективних, практико-орієнтованих курсів), що враховує індивідуально-особистісні можливості і специфічні властивості учнів.

Індивідуальний освітній маршрут (шлях) визначається освітніми вимогами, індивідуальними здібностями, а також можливостями студента (рівень готовності до освоєння програм), а також існуючими стандартами змісту освіти. Індивідуальний освітній маршрут представляє змістовний компонент, який відповідає на питання «що має змінитися в особистості в ході реалізації даного маршруту?».

Ф.Г. Мухаметзянової, Р.В. Забирова [9] наголошують на тому, що індивідуальний освітній маршрут – це проект індівідуально-

диференційованого освоєння основної освітньої програми, що забезпечує студенту вузу позицію суб'єкта вибору індивідуальної освітньої траєкторії, розробки та реалізації основної освітньої програми при реалізації педагогічних умов. Участь студента в проектуванні індивідуальної освітньої траєкторії залежить від властивостей його суб'єктності: мотивації; цілеспрямованості, планування, прийняття рішення, безконфліктного спілкування, комунікативності, самовизначення, рефлексії.

Спільна діяльність процесу проектування індивідуальної освітньої траєкторії і маршруту повинна здійснюватися в певній послідовності:

– педагог і студент ВНЗ як суб'єкти освітньої діяльності вибудовують, обговорюють, доповнюють уявлення про індивідуальну освітню траєкторії студента;

– студент (суб'єкт навчально-професійної діяльності) і педагог поєднують свої уявлення і розробляють модель ІОТ;

– педагог засвідчується в тому, що студент розуміє сенс і приймає на себе відповідальність за результат вибору і реалізації ІОТ;

– в процесі спільної навчально-професійної діяльності відбувається коригування індивідуальної освітньої траєкторії і маршруту студента ВНЗ;

– при наявності проблемних ситуацій, які студент не може вирішити самостійно, він звертається до педагога;

– підсумком реалізації ІОТ є успішне освоєння і засвоєння студентом основної освітньої програми ВНЗ.

Науковці Н.М. Уварова та Т.В. Максимченко [10] вважають, що проектування індивідуальної освітньої програми (ІОП) здійснюється на основі взаємодії того, хто навчається і педагогів, передбачає тісну співпрацю і співтворчість. Однак ІОП не може бути сформована раз і назавжди. Змінюються запити учнів, змінюються і зовнішні умови організації освітнього процесу, з'являються нові освітні ресурси. Нормативно-методичне

забезпечення ІОП має передбачати механізми її корекції та механізми її супроводу. На сьогодні, одним із дієвих механізмів супроводу ІОП є розробка програм тьюторської підтримки. Така програма відображає три пласти тьюторського супроводу: допомога, спрямована на розвиток автономності і самостійності суб'єкта навчання; фасилітація (полегшення взаємодії студента з всіма учасниками освітнього процесу); супровід у вирішенні проблем в професійному становленні. Дослідження етимології терміна «супровід» дозволяє розглядати його в контексті тьюторської діяльності як соціальну взаємодію, функціями якого є розвиток учня в різних навчальних, соціальних, особистісних ситуаціях за допомогою спеціальних засобів (тьюторських технологій). Цілеспрямований розвиток особистості супроводжуваного через створення умов для успішного навчання, професійного і саморозвитку – провідна ідея тьюторського супроводу студента.

Провідними умовами для реалізації ІОП повинні стати:

– розробка і реалізація в освітньому процесі програм різного рівня і спрямованості, створення надлишкового середовища ресурсів (освітніх програм);

– забезпечення спадковості змісту початкової, середньої та вищої професійної освіти;

– розвиток механізмів взаємодії ВНЗ з роботодавцями, соціальними партнерами;

– реалізація моделей психолого-педагогічної та тьюторської підтримки індивідуальних освітніх траєкторій студентів;

– інформаційний і технологічний розвиток освітнього середовища, розвиток різнорівневих інформаційних ресурсів;

– розвиток системи моніторингу особистісного просування студента.

О.В. Гончарова та Р.М. Чумичева [11] стверджують, що впровадження індивідуально-орієнтованої організації навчального процесу, що стимулює

бально-рейтингову систему оцінки навчальної діяльності, яка нерозривно пов’язана з асинхронною організацією навчального процесу. З урахуванням зарубіжної та вітчизняної термінології всі типи організації навчального процесу пропонується розділити на два класи: синхронні і асинхронні. Найбільш часто під асинхронним навчанням розуміють таку форму організації навчального процесу, яка забезпечує студенту можливість освоєння навчальних матеріалів у будь-який час, який є для нього зручним, та не встановлюється заздалегідь розкладом занять. Студенти потребують саме асинхронного навчання.

Для впровадження пропонованої ними моделі огранізації асинхронного навчання необхідно:

– забезпечення компетентних консультацій тьюторів;

– нормування самостійної роботи студентів (розробка і впровадження єдиної бально-рейтингової системи оцінок і технологічних карт);

– оперативний, об’єктивний, прозорий і постійний облік результатів навчання.

У таблиці 1.1 наведено короткий опис характерних особливостей наявних підходів.

Таблиця 1.1 – Характеристика наявних підходів

Назва	Опис
Система «ТРАЕКТОРИЯ.ОНЛАЙН»	Система, де школярі і студенти можуть безкоштовно пройти тестування та дізнатися, які навчальні заклади та місця роботи їм підходять з урахуванням психологічних і особистісних характеристик. Ціль системи: створити умови для усвідомленого вибору індивідуальних освітніх траєкторій і побудови успішної кар’єри школярів і студентів
Система Dean-Dashboard.com	Це сайт, який допомагає орієнтуватися в джерелах електронного навчання. В ньому систематизована інформація про освіту онлайн. Система допомагає

Продовження таблиці 1.1

Назва	Опис
	будувати власний навчальний план відповідно до особистих можливостей і цілей Ціль системи: допомогти людям побудувати особистий шлях навчання з використанням безкоштовних онлайн-курсів, як альтернативу традиційній вищій освіті
Система в Тюменському університеті	Студенти університету, які навчаються за новою освітньою моделлю, можуть самостійно формувати свій власний індивідуальний навчальний план за рахунок вибору навчальних курсів (понад 200 елективів) з п’яти областей знань поряд з вивченням обов’язкових дисциплін Дисциплінарна широта і висока варіативність програми в загальноосвітній і професійної частинах дозволяють отримати унікальний набір компетентностей. Можливість вибору додаткового профілю підвищує привабливість випускників на ринку праці, а мотивація студентів підвищується
Дослідження А.В. Проворової	Вчена вважає, що специфіка побудови індивідуального освітнього маршруту в проявляється у великих можливостях вибору профільних занять (елективних, практико-орієнтованих курсів), що враховує індивідуально-особистісні можливості і специфічні властивості учнів На стадії реалізації програми студент виступає як суб’єкт здійснення освіти. В цьому випадку особистісно орієнтований освітній процес реалізується як індивідуальний освітній маршрут при використанні функціональних можливостей педагогічної підтримки

Продовження таблиці 1.1

Назва	Опис
Дослідження Ф.Г. Мухаметзянової та Р.В. Забирова	Науковці наголошують на тому, що участь студента в проектуванні індивідуальної освітньої траєкторії залежить від мотивації, цілеспрямованості, планування, прийняття рішення, безконфліктного спілкування, комунікативності, самовизначення. Також наголошують на тому, що для реалізації освітніх траєкторій необхідно дотримуватися таких
Дослідження Ф.Г. Мухаметзянової та Р.В. Забирова	груп педагогічних умов: організаційно-педагогічних, навчально-методичних та психолого-педагогічної підтримки
Дослідження Н.М. Уварової та Т.В. Максимченко	Вчені кажуть, що проектування індивідуальної освітньої траєкторії передбачає тісну співпрацю і співтворчість педагогів та студентів. Також вони вважають, що траєкторія не може бути сформована раз і назавжди, адже запити студентів, змінюються, з'являються нові освітні ресурси та інше Тьюторську підтримку вважають дуже доречною, вважають, що її ідея – цілеспрямований розвиток особистості супроводжуваного через створення умов для успішного навчання, професійного і саморозвитку
Ідея О.В. Гончарової та Р.М. Чумичевої	Науковці вважають, що при побудові індивідуальних траєкторій необхідно забезпечити компетентні консультації тьюторів. Також вважають за необхідне запровадити єдину рейтингову систему оцінок, а також ведення оперативного та прозорого обліку результатів навчання

Як бачимо, усі програми мають схожі функції та ідею для формування індивідуальної освітньої траєкторії, але жодна з них не охоплює велику кількість факторів впливу на формування траєкторії, одні мають ідею

формування лише опираючись на невелику кількість даних про претендента, інші цілком роблять відповідальними за вибір тільки самого претендента, треті надають велику кількість повноважень тьюторам.

1.3 Постановка задачі

1.3.1 Призначення розробки

Призначенням розробки є спрощення створення індивідуальних освітніх траєкторій для студентів вищих навчальних закладів, в яких враховуються психологічні, професійні та особисті якості претендентів.

1.3.2 Мета та задачі розробки

Головною метою розробки є покращення якості навчання студентів за рахунок створення їх індивідуальних освітніх траєкторій.

Головну мету можна декомпозувати на підцілі. Декомпозиція необхідна для побудови дерева цілей, щоб пов’язати головну мету зі способами її досягнення, що сформовані у вигляді задач окремим виконавцям. Підцілі проекту можна сформулювати так:

– підвищення мотивації студентів закладів вищої освіти (ЗВО) до навчання;

– покращення професійної орієнтації студентів ЗВО;

– підвищення рівня професійної компетентності студентів ЗВО.

Для досягнення поставлених підцілей мають бути вирішені такі задачі:

– автоматизація виявлення особистісних якостей студентів на основі психологічних тестів;

– формування психологічного портрету на основі визначених лідерських якостей роботи в команді і його супровід у БД;

– визначення профпридатності студента шляхом анкетування;

– контент-аналіз мотиваційних листів чи есе студентів за допомогою алгоритмів NLP-аналізу тексту;

– виявлення професійних компетентностей (знання, уміння, досвід) претендента;

– кореляційний аналіз професійних компетентностей претендента та компетенцій за IT-професіями;

– кореляційний аналіз компетенцій IT-професій та навчальних дисциплін;

– розробка автоматизованих процедур компоновки персонального навчального плану;

– формування звітності.

Висновок до розділу

У ході написання даного розділу були наведені загальні відомості про предметне середовище: актуальність проблеми, її необхідність та можливі шляхи освіти. Було визначено недоліки стандартної форми навчання та запропоновано альтернативи типовому навчальному процесу: висвітлено ідею індивідуальних освітніх траєкторій. Також було виявлено переваги персоналізованої системи освіти для усіх учасників навчального процесу.

Послідовність дій для процесу побудови індивідуальної траєкторії представлена за допомогою діаграми діяльності. Функціональна модель була описана за допомогою контекстної діаграми та діаграму процесу в нотації IDEF0.

Розглянуто аналоги системи: система «ТРАЕКТОРИЯ.ОНЛАЙН», Dean-Dashboard.com, а також система побудови індивідуальних траєкторій в Тюменському університеті. Проведено аналіз публікацій різних науковців та виявлено відмінності між ідеями визначених статей та вже розроблених

аналогів, тим самим виявлено, що кожен з аналогів охоплює невелику кількість факторів впливу на формування траєкторії.

Було визначено постановку задачі – призначення, головну мету, яку було декомпозовано на підцілі. Для досягнення підцілей були визначені задачі для вирішення.

2 ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ

2.1 Вхідні дані

Вхідні дані системи формування індивідуальної освітньої траєкторії надходять з декількох джерел, а саме від:

- претендентів;
- стейкхолдерів;
- тьюторів.

Тепер детально розглянемо, які саме дані надходять з цих джерел.

Дані, які надходять від претендентів. На початку формування індивідуальної траєкторії претендент повинен подати заявку та повідомити особисті дані: прізвище, ім’я, електронну пошту.

А безпосередньо перед здійсненням формування претендент має надати доступ до своїх результатів тестів та власних побажань.

Дані, які надходять від стейкхолдерів. При пошуку компетентностей претендентів та формуванні списку професій стейкхолдер повинен пред’явити інформацію про себе: назву компанії, в якій працює, профіль компанії, свою посаду, а також повідомити свої вимоги: потрібні компетентності в контексті професій.

Дані, які надходять від тьюторів. Тьютори вказують своє прізвище, ім’я, по-батькові, а також свою область діяльності.

2.2 Вихідні дані

Вихідними даними є сформована індивідуальна освітня траєкторія, яка генерується системою. У таблиці 2.1 наведений приклад макету траєкторії.

Таблиця 2.1 – Індивідуальна освітня траєкторія

Напрям	**ПІБ**	**Назва траєкторії**	**Курс**
Назва дисципліни		Число кредитів	Форма контролю
Всього за рік			
Підпис претендента _____			
Підпис тьютора _____			

2.3 Опис структури бази даних

Опишемо структуру бази даних системи формування індивідуальної траєкторії у вигляді діаграми ERD (рисунок 2.1). Фізична модель бази даних системи наведена у графічному матеріалі.

Було виділено такі сутності системи:

- користувач;
- претендент;
- стейкхолдер;
- тьютор;
- професія;
- компетентність;
- компетентність у професії;
- дисципліна;
- дисципліна для претендента;
- область знань;
- професія в області знань;

– дисципліна в області знань;

– навчальний план.

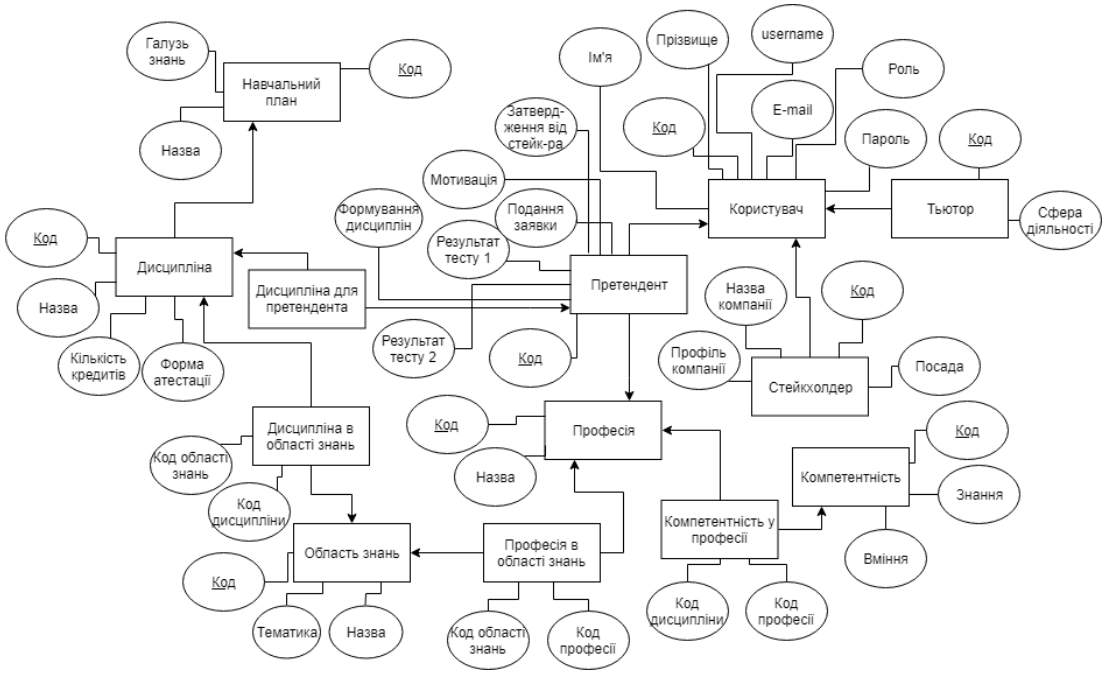


Рисунок 2.1 – Схема структурна бази даних системи у вигляді діаграми ERD

У таблиці 2.2 наведено зв’язки між сутностями.

Таблиця 2.2 – Зв’язки між сутностями системи

Перша сутність	**Друга сутність**	**Тип зв’язку**
Користувач	Претендент	«один до багатьох»
	Стейкхолдер	
	Тьютор	
Претендент	Професія	«багато до одного»
Професія	Компетентність	«багато до багатьох»
Професія	Область знань	«багато до багатьох»
Область знань	Дисципліна	«багато до багатьох»
Дисципліна	Претендент	«багато до багатьох»
Дисципліна	Навчальний план	«багато до одного»

Як бачимо з таблиці 2.2, між деякими сутностями професією та компетентністю існує зв’язок «багато до багатьох». У таблиці 2.3 показано проміжні сутності, які вводяться для розірвання цих зв’язків.

Таблиця 2.3 – Проміжні сутності для розірвання зв’язків типу «багато до багатьох»

Перша сутність	**Друга сутність**	**Проміжна сутність**
Професія	Компетентність	Компетентність у професії
Професія	Область знань	Професія в області знань
Область знань	Дисципліна	Дисципліна в області знань
Дисципліна	Претендент	Дисципліна для претендента

Базу даних реалізовано за допомогою модуля SQLite. Це автономний та транзакційний механізм бази даних SQL, що працює без сервера.

Далі у таблицях 2.4 – 2.16 наведено опис таблиць розробленої бази даних.

Таблиця 2.4 – Таблиця користувачів

Назва таблиці	**Назва стовпця**	**Тип даних**	**Детальна інформація**
app_user – таблиця користувачів системи	id	integer	Код, під яким користувач зареєстрований в базі даних
	name	varchar(20)	Ім’я користувача
	surname	varchar(20)	Прізвище користувача
	password	varchar(20)	Пароль користувача
	role	varchar(10)	Роль користувача
	useremail	varchar(254)	E-mail корисувача
	username	varchar(20)	username користувача

Таблиця 2.5 – Таблиця претендентів

Назва таблиці	**Назва стовпця**	**Тип даних**	**Детальна інформація**
app_applicant – таблиця претендентів системи	id	integer	Код, під яким претендент зареєстрований в базі даних

Продовження таблиці 2.5

Назва таблиці	Назва стовпця	Тип даних	Детальна інформація
	test1	integer	Результат першого тесту
	test2	integer	Результат другого тесту
	motivation	boolean	Наявність мотивації
	appliedForTraj	boolean	Подання заявки
	approvedTraj	boolean	Формування дисциплін тьютором
	approvedStake	integer	Затвердження заявки від стейкхолдера (0 – затвердження очікується, 1 – затверджено, -1 – не затверджено)
	userid	integer	Код, під яким користувач зареєстрований в базі даних
	jobid	integer	Код, під яким професія зареєстрована в базі даних

Таблиця 2.6 – Таблиця стейкхолдерів

Назва таблиці	Назва стовпця	Тип даних	Детальна інформація
app_stakeholder– таблиця стейкхолдерів	id	integer	Код, під яким стейкхолдер зареєстрований в базі даних
	companyName	varchar(20)	Назва компанії, в якій працює стейкхолдер
	scopeCompany	varchar(20)	Профіль компанії

Продовження таблиці 2.6

Назва таблиці	Назва стовпця	Тип даних	Детальна інформація
	position	varchar(20)	Позиція стейкхолдера
	userid	integer	Код, під яким користувач зареєстрований в базі даних

Таблиця 2.7 – Таблиця тьюторів

Назва таблиці	Назва стовпця	Тип даних	Детальна інформація
app_tutor– таблиця тьюторів	id	integer	Код, під яким стейкхолдер зареєстрований в базі даних
	scope	varchar(30)	Сфера діяльності тьютора
	userid	integer	Код, під яким користувач зареєстрований в базі даних

Таблиця 2.8 – Таблиця професій

Назва таблиці	Назва стовпця	Тип даних	Детальна інформація
app_job– таблиця професій	id	integer	Код, під яким професія зареєстрована в базі даних
	name	varchar(20)	Назва професії

Таблиця 2.9 – Таблиця компетентностей

Назва таблиці	Назва стовпця	Тип даних	Детальна інформація
app_competence– таблиця компетентностей	id	integer	Код, під яким компетентність зареєстрована в базі даних
	knowledge	varchar(1000)	Знання
	skill	varchar(1000)	Вміння

Таблиця 2.10 – Таблиця компетентності у професії

Назва таблиці	Назва стовпця	Тип даних	Детальна інформація
app_compjob– таблиця компетентності у професії	id	integer	Код, під яким компетентність у професії зареєстрована в базі даних
	competenceid	integer	Код, під яким компетентність зареєстрована в базі даних
	jobid	integer	Код, під яким професія зареєстрована в базі даних

Таблиця 2.11 – Таблиця дисциплін

Назва таблиці	Назва стовпця	Тип даних	Детальна інформація
app_discipline– таблиця дисциплін	id	integer	Код, під яким дисципліна зареєстрована в базі даних
	name	varchar(50)	Назва дисципліни
	credits	float	Кількість кредитів
	formOfControl	varchar(20)	Форма контролю

Продовження таблиці 2.11

Назва таблиці	Назва стовпця	Тип даних	Детальна інформація
	curriculumid	integer	Код, під яким навчальний план зареєстрований в базі даних

Таблиця 2.12 – Таблиця дисципліни для претендента

Назва таблиці	Назва стовпця	Тип даних	Детальна інформація
app_disciplinefor applicant – таблиця дисципліни для претендента	id	integer	Код, під яким дисципліна для претендента зареєстрована в базі даних
	disciplineid	integer	Код, під яким дисципліна зареєстрована в базі даних
	applicantid	integer	Код, під яким претендент зареєстрований в базі даних

Таблиця 2.13 – Таблиця області знань

Назва таблиці	Назва стовпця	Тип даних	Детальна інформація
app_knowledgearea – таблиця області знань	id	integer	Код, під яким область знань зареєстрована в базі даних
	name	varchar(50)	Назва
	topics	varchar(500)	Тематика області знань

Таблиця 2.14 – Таблиця професії в області знань

Назва таблиці	Назва стовпця	Тип даних	Детальна інформація
app_jobsinareas – таблиця професії в області знань	id	integer	Код, під яким професія у області знань зареєстрована в базі даних
	jobid	integer	Код, під яким професія зареєстрована в базі даних
	areaid	integer	Код, під яким область знань зареєстрована в базі даних

Таблиця 2.15 – Таблиця дисципліни в області знань

Назва таблиці	Назва стовпця	Тип даних	Детальна інформація
app_jobsinareas – таблиця дисципліни в області знань	id	integer	Код, під яким професія у області знань зареєстрована в базі даних
	disciplineid	integer	Код, під яким дисципліна зареєстрована в базі даних
	areaid	integer	Код, під яким область знань зареєстрована в базі даних

Таблиця 2.16 – Таблиця навчальних планів

Назва таблиці	Назва стовпця	Тип даних	Детальна інформація
app_curriculum– таблиця навчальних планів	id	integer	Код, під яким навчальний план зареєстрований в базі даних
	name	varchar(50)	Назва
	field	varchar(20)	Галузь знань

Висновок до розділу

У ході написання даного розділу було визначено вхідні та вихідні дані до системи побудови індивідуальної освітньої траєкторії. Вхідні дані надходять з декількох джерел, а саме від: претендентів, стейкхолдерів та тьюторів. Вихідними даними є сформована індивідуальна освітня траєкторія. У розділі було наведено табличний опис (макет) вихідного документу.

Також було описано структуру даних системи. Структурами даних у системі побудови індивідуальної траєкторії є база даних. Для опису структури бази даних було використано ER-діаграму. У ER-моделі відображається функціональна залежність атрибутів сутностей, а також зв’язки між ними. Опис зв’язків було подано у табличному вигляді.

У цьому розділі було наведено детальний опис структури бази даних у вигляді таблиць, які містять інформацію про назву таблиці, назви стовпців таблиці, типи даних стовпців, а також їх призначення.

3 МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ

3.1 Змістовна постановка задач

Припустимо є абітурієнт, який ще не визначився з професією і має можливість надіслати своє есе або мотиваційний лист. У зв’язку з цим постає необхідність чіткого структурування відповідей претендента відповідно до певної тематичної рубрики. Це дозволяє як абітурієнту, так і викладачам отримати доступ до інформації про те, чи підходить майбутній студент напрямку навчання на певному факультеті, а також визначити тематику написаного ним тексту для повного аналізу його намірів.

Таким чином, проаналізувавши його есе чи мотиваційний лист, можна буде прийти до висновку, чи є він вмотивованим студентом, чи ні. Якщо текст має тематику ІТ-сфери чи якимось чином відноситься до неї, то вважається, що студент розуміє, чого він хоче та впевнений у своїх цілях.

Для аналізу введеної інформації використовується метод латентно-семантичного аналізу, який дозволяє автоматично проаналізувати вміст текстової інформації, що міститься в документах, і виявляти приховані семантичні (смислові) зв’язки між документами [12].

Тепер уявімо, що студент вже визначився з професію, пройшовши тест на професійні компетентності. Далі він прагне обрати дисципліни для формування індивідуальної траєкторії.

Для цього спочатку введемо поняття області знань. Сукупність знань з інформатики організована у вигляді ієрархічної структури. На верхньому рівні ієрархії знаходиться область знань – це окрема частина дисципліни інформатики. Кожна область поділяється на менші структури, які представляють собою окремі тематичні модулі всередині області знань.

Наприклад, маємо область знань «Алгоритми та їх складність». Її тематичні модулі (тематики):

– базовий аналіз;

– алгоритмічні стратегії;

– основи структур даних та алгоритмів;

– основи теорії автоматів, обчислюваність та складність;

– обчислювальна складність розширеного рівня;

– розширена теорія автоматів і обчислюваність;

– розширені структури даних, алгоритми та аналіз.

Для того, щоб дізнатися дисципліни для певної професії потрібно пов’язати професію з областями знань, таким чином визначивши набір областей знань для кожної професії. Визначивши області знань, можемо визначити і дисципліни, витягнувши ключові слова із списку областей знань для певної професії. Ключові слова будуть показані тьюторові, який і визначить набір дисциплін для професії, опираючись на ключові слова.

Виникає питання, як пов’язати професії та області знань, а також як визначити ключові слова для областей знань.

Для того, щоб пов’язати професії та області знань використовується метод латентно-семантичного аналізу, який дозволяє автоматично проаналізувати компетентності професій, які подано у текстовому вигляді, та тематики областей знань (також подано у вигляді тексту), і виявляти приховані семантичні (смислові) зв’язки між компетентностями та тематиками. Таким чином буде знайдено список областей знань для кожної професії.

Для виявлення ключових слів областей знань достатньо знайти ключові слова їх тематик. Ключові слова – тематичні рубрики тематик.

Таким чином, маємо тематики у вигляді тексту, який потрібно розбити на тематичні рубрики.

Отже, визначимо задачі для вирішення:

– задача перевірки вмотивованості студента;

– задача визначення дисциплін.

3.2 Математична постановка задачі

3.2.1 Задача перевірки вмотивованості студента

Нехай дана колекція з n документів, набір яких представляє собою есе, чи мотиваційний лист абітурієнта, і m різних термінів, видобутих із словника IT-термінів. Під терміном розуміється або окреме слово, або словосполучення. Для застосування моделі латентно-семантичного аналізу, потрібно побудувати $m \times n$ матрицю «термін-документ» X, з комірками $x_{i,j}$, що містять вагові коефіцієнти терміна t_i, в документі d_j. Стовпці матриці X на практиці відповідають мультимножині слів (англ. «bag-of-words») для документа, при цьому можуть бути використані терміни, «зважені» за якою-небудь схемою: наприклад, це може бути широко відома схема TF-IDF (від англ. TF – term frequency, IDF – inverse document frequency) або також можуть бути ефективні різні схеми, засновані на ентропії.

Отримана матриця «термін-документ» X являє собою просторово-векторну модель представлення текстової інформації і одночасно вхідні дані для методу латентно-семантичного аналізу [13].

Потрібно дати відповідь на питання, чи відповідає вхідний текст тематиці IT-сфери та визначити тему цього тексту, застосовуючи методи семантичного аналізу тексту.

3.2.2 Задача визначення дисциплін

Розділимо дану задачу на підзадачі:

– визначення списку областей знань для професії;

– визначення ключових слів з тематик обраних областей знань.

3.2.2.1 Підзадача визначення списку областей знань для професії

Використаємо математичну постановку задачі визначення вмотивованості студента.

Нехай дана колекція з n документів, які являють собою компетентності професії, і m різних термінів, видобутих із тематик усіх областей знань. Для застосування моделі латентно-семантичного аналізу, потрібно побудувати $m \times n$ матрицю «термін-документ» X, з комірками $x_{i,j}$, що містять вагові коефіцієнти терміна t_i, в документі d_j.

Отримана матриця «термін-документ» X являє собою просторово-векторну модель представлення текстової інформації і одночасно вхідні дані для методу латентно-семантичного аналізу [13].

Для визначення дисциплін потрібно дати відповідь на питання, як пов'язані професії та області знань, а також визначити області знань для кожної професії.

3.2.2.2 Підзадача визначення ключових слів із тематик обраних областей знань

Нехай D – множина текстових документів, які являють собою тематики області знань, W – множина термінів, що вживаються в них. Кожен документ $d \in D$ представлений послідовністю термінів $\{w_i\}_{i=1}^{n_d}$ із W, де n_d – число слів у документі d. Один і той же термін може зустрічатися в документі кілька разів.

Нехай Z – це скінченна множина тематик. Нехай поява терміна w в кожному документі d пов'язана з деякою невідомою, тематикою $z \in Z$. Користуючись цим, уявімо безліч документів у вигляді безлічі трійок вигляду (d, w, z), обраних випадково і незалежно з дискретного розподілу $p(d, w, z)$, яке задано на множині $D \times W \times Z$. Незалежність елементів вибірки має на увазі те, що порядок термінів в документі не важливий для виявлення тематик.

Завдання імовірнісного тематичного моделювання можна визначити наступним чином: побудувати імовірнісну тематичну модель для колекції документів D, тобто визначити множину тематик Z, ймовірність появи терму

в певній темі – $p(w|z)$ для всіх тематик $z \in Z$ і ймовірність появи теми в документах – $p(z|d)$ для всіх документів $d \in D$ [14].

3.3 Обґрунтування методів розв'язання

Латентно-семантичний аналіз відображає документи і окремі слова в так званий «семантичний простір», у якому відбуваються всі подальші порівняння. При цьому робляться такі припущення:

– документи це просто набір слів. Порядок слів у документах ігнорується. Важливо тільки те, скільки разів те чи інший слово зустрічається в документі;

– семантичне значення документа визначається набором слів, які як правило йдуть разом. Наприклад, в біржових зведеннях, часто зустрічаються слова: «фонд», «акція», «долар»;

– кожне слово має єдине значення.

Латентний розподіл Діріхле (LDA) – найпоширеніша тематична модель, яка використовується в даний час.

Тематичні моделі, дозволяють документу або терміну відноситися відразу до кількох тем з різними можливостями. Вони описують кожну тему дискретним розподілом на множині термінів, кожен документ – дискретним розподілом на множині тем. Передбачається, що колекція документів – це послідовність термінів, обраних випадково і незалежно з суміші таких розподілів, і ставиться завдання відновлення компонент суміші за вибіркою.

Завдання тематичного моделювання – це знаходження невідомих, прихованих, матриць ймовірностей розподілу слів за темами і тем по документам [15].

Для реалізації алгоритму класифікації в головному алгоритмі семантичного аналізу найбільш ефективною виявилась схема TF-IDF.

TF – частота слова (англ. Term frequency) – відношення числа входжень деякого терміну до загальної кількості слів документа. Таким чином, оцінюється важливість терміна в межах окремого t_iдокумента.

$$tf(t,d) = \frac{n_t}{\sum_k n_k},$$

де n_t – це число входжень терміна t в документ d, а у знаменнику – загальна кількість слів у даному документі.

IDF – зворотна частота документа (англ. Inverse document frequency) – інверсія частоти, з якою окреме слово зустрічається в документах загальної колекції. Обчислення зворотної частоти документа зменшує вагу слів широкого вживання. Для кожного унікального слова в межах конкретного набору документів існує тільки лише значення IDF.

$$idf(t,D) = log\frac{|D|}{|\{d_i \in D \mid t \in d_i\}|},$$

де $|D|$ – число документів в колекції, а $|\{d_i \in D \mid t \in d_i\}|$ – число документів із колекції D,в яких зустрічається t (коли $n_t \neq 0$).

Вибір основи логарифма в формулі не має значення, оскільки зміна основи призводить до зміни ваги кожного конкретного слова на сталий множник, що не впливає на співвідношення ваг.

Отже, міра TF-IDF – це добуток двох співмножників:

$$tfidf\ (t, d, D) = tf\ (t, d) \times idf\ (t, D)$$

Велику вагу в TF-IDF отримуватимуть слова з високою частотою в межах визначеного документа і з низькою частотою використань в інших документах [13].

Отже, для рішення задачі перевірки вмотивованості було обрано метод латентно-семантичного аналізу, а для реалізації класифікатора – схема TF-IDF. Для задачі визначення дисциплін було також обрано метод латентно-семантичного аналізу (підзадача визначення списку областей знань для професії) та метод латентного аналізу Діріхле (підзадача визначення ключових

слів із тематик обраних областей знань) Ці два методи відомі дуже високою якістю рішень, саме тому було вирішено їх обрати.

3.3.1Латентно-семантичний аналіз

Дано: колекція з n документів і m різних термінів.

Знайти: відповідність документів заданим термінам.

Після побудови матриці «термін-документ» застосовується сингулярне розкладання (англ. Singular value decomposition, SVD) – математична методика, яка обчислює розкладання вихідної матриці X на три матриці.

$$X = USV^t$$

З отриманих матриць, матриці U і V є ортогональними матрицями розмірності $m \times r$ і $n \times r$ відповідно, а матриця S – це $r \times r$ діагональна матриця, яка містить власні значення. Обгрунтування полягає в тому, що кожне з r власних значень відповідає одному з вищезазначених високорівневих компонентів, що відслідковуються в колекції документів, і позначає, наскільки цей компонент актуальний у всій колекції.

Власні значення упорядковано відповідно до діагоналі матриці S в порядку спадання, так що ті власні значення, які йдуть першими, пов’язані з найбільш важливими компонентами. Це дозволяє легко відрізати найменш важливі компоненти до числа $k \leq r$, просто видаливши відповідні рядки і стовпці в матрицях. Це скорочення потенційно дозволяє видалити «шум» в даних, який може бути складений, наприклад, з термінів або груп, що з’являються тільки в декількох документах і погано пов’язані з іншими.

Як тільки таке значення k встановлено, можна розрахувати побудовану апроксимувати версію вихідної матриці «термін-документ» X шляхом перемноження трьох усічених матриць: результуюча матриця X' матиме свій ранг, зменшений від r до k. Матриця X' структурно ідентична матриці X (її рядки і стовпці є уявленнями для тих же термінів і документів, що і в матриці

X), але вагові коефіцієнти скориговані тепер так, що «шум» усунутий, і враховано очевидні взаємозв'язки між термінами (або між документами). Наприклад, якщо два терміни t_a і t_b часто зустрічаються разом в документах, то документ, який містить тільки термін t_a з цих двох термінів, буде в будь-якому випадку мати вагу для терміна t_b більше нуля (і навпаки).

З відновленої матриці X' або безпосередньо з усічених матриць, використовуваних для її обчислення, схожість між термінами і між документами може бути обчислена відповідно до скоригованих вагових коефіцієнтів, які в загальному випадку будуть відрізнятися від відповідних ваг, обчислених з вихідної матриці. У загальному випадку, коли повинні бути знайдені документи, які найбільше задовольняють запиту, використовується загальний підхід, в якому запит задається, як документ, який повинен бути порівняний або зіставлений з якимось відомим документом. Він повинен бути спочатку відображений в прихований (латентний) простір для того, щоб пройти таку ж корекцію значень: ця процедура відома як згортка. У прихованому просторі можна знайти пов'язані документи, які не містять точних слів запиту, але при цьому строго відповідають їм [13].

3.3.2Латентне розміщення Діріхле

Інший алгоритм для рішення підзадачі визначення ключових слів із тематик обраних областей знань – латентне розміщення Діріхле.

Дано: D – множина текстових документів, які являють собою тематики області знань, W – множина термінів, що вживаються в них, Z – це скінченна множина тематик.

Знайти: множину тематик Z.

Розглянемо породжуючу імовірнісна модель. У цій моделі використовується гіпотеза про умовну незалежності, яка вказує на те, що ймовірність появи терміна w за умови того, що обрана тематика w, описується

розподілом $p(w|z)$ і не залежить від документа d. Це еквівалентно наступним рівностям:

$$p(w|d,z) = p(w|z),$$
$$p(d,w|z) = p(d|z)p(w|z).$$

Використовуючи гіпотезу умовної незалежності і визначення умовної і повної ймовірності, отримуємо:

$$p(w|d) = \sum_{z\in Z} p(w|z)p(z|d).$$

Дана рівність описує процес породження безлічі документів D, якщо відомі розподіли (ймовірність) $p(w|z)$ і $p(z|d)$. Процес побудови тематичної моделі є зворотною задачею і пов'язаний з пошуком розподілів $p(w|z)$ і $p(z|d)$ за відомою множиною документів D. Адже з іншого боку:

$$\frac{n_{dw}}{n_d} \approx p(w|d),$$

де n_{dw} – кількість входжень слова w в документ d, n_d – розмір документу.

Роглянемо тематичну модель латентного розміщення Діріхле.

Введемо наступні означення:

$$\Phi = (\varphi_{wz})_{|W|\times|Z|}, \quad \varphi_{wz} = p(w|z),$$
$$\theta = (\vartheta_{zd})_{|Z|\times|D|}, \quad \vartheta_{zd} = p(z|d),$$

де Ф – матриця тематик термінів, а θ – матриця тематик документів.

Матриці Ф та θ – стохастичні. Стохастична матриця – це матриця з нормованими стовпцями та невід'ємними елементами.

Модель латентного розміщення Діріхле заснована на ймовірнісній моделі появи пари «документ-слово», яка може буде представлена наступним чином:

$$p(d,w) = \sum_{z\in Z} p(w|z)p(z|d)p(d),$$

де $p(d)$ – це апріорний розподіл на множині документів.

Далі у методі латентного розміщення Діріхле робиться припущення, що вектори документів $\vartheta_d = (\vartheta_{zd}) \in R^{|Z|}$ і вектори тематик $\varphi_z = (\varphi_{wz}) \in R^{|W|}$

породжуються розподілом Діріхле з параметрами $\alpha \in R^{|Z|}$ і $\beta \in R^{|W|}$ відповідно:

$$Dir(\vartheta_d; \alpha) = \frac{\Gamma(\alpha_0)}{\prod_z \Gamma(\alpha_z)} \prod_z \vartheta_{zd}^{\alpha_z - 1}, \alpha_z > 0, \alpha_0 = \sum_z \alpha_z, \vartheta_{zd} = 1;$$

$$Dir(\varphi_z; \beta) = \frac{\Gamma(\beta_0)}{\prod_w \Gamma(\beta_w)} \prod_w \varphi_{wz}^{\beta_w - 1}, \beta_w > 0, \beta_0 = \sum_w \beta_w, \varphi_{wz} = 1,$$

де $\Gamma(z)$ – гамма-функція.

З огляду на дані припущення, розглянемо Байєсову мережу моделі латентного розміщення Діріхле, зображену на рисунку 3.1.

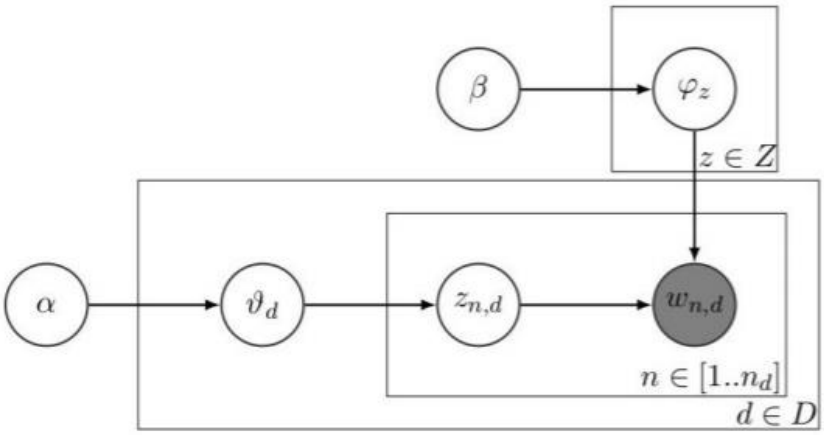


Рисунок 3.1 – Байєсова мережа моделі латентного розміщення Діріхле

Параметри α і β є гіперпараметрами моделі і одночасно параметрами розподілу Діріхле, і, як правило, задаються до початку навчання моделі. Змінна $w_{n,d}$ є змінною спостерення і являє собою термін, що стоїть на n-ій позиції в документі d. Решта всіх змінних є латентними (прихованими).

Для оцінки параметрів моделі латентного розіщення Діріхле по колекції документів d застосовуються варіаційний Байєсівський висновок або метод семплювання Гіббса [14].

Якщо потрібно, щоб система була ефективною і масштабувалася при постійно зростаючому корпусі, то найчастіше застосовують латентне розміщення Діріхле (LDA). LDA є генеративною моделлю, здатною породжувати нові документи. У LDA кожен документ може розглядатися як набір різних тем [16].

3.4 Опис методів розв'язання

3.4.1. Латентно-семантичний аналіз

Розглянемо схему алгоритму латентно-семантичного аналізу [17].

Крок 0. З аналізованих документів ВИКЛЮЧИТИ стоп-слова. Це слова, які зустрічаються в кожному тексті і не несуть в собі смислового навантаження, це, перш за все, всі сполучники, частки, прийменники і безліч інших слів. *ПЕРЕЙТИ НА* Крок 1

Крок 1. З аналізованих документів необхідно ВІДФІЛЬТРУВАТИ цифри, окремі букви і розділові знаки. *ПЕРЕЙТИ НА* Крок 2

Крок 2. З усіма словами з документів повинна бути проведена операція стемінг – отримання основи слова. *ПЕРЕЙТИ НА* Крок 3

Крок 3. СКЛАСТИ частотну матрицю індексованих слів. У цій матриці стовпці відповідають документам, а рядки – індексованим словами. У кожному осередку матриці має бути зазначено, яку кількість разів слово зустрічається у відповідному документі. *ПЕРЕЙТИ НА* Крок 4

Крок 4. Отриману частотну матрицю слід НОРМАЛІЗУВАТИ за допомогою способу нормалізації матриці TF-IDF. *ПЕРЕЙТИ НА* Крок 5

Крок 5. ПРОВЕСТИ сингулярне розкладання отриманої матриці. *ЯКЩО* використовується двовимірне сингулярне розкладання *ПЕРЕЙТИ НА* Крок 6

Крок 6. ВІДКИНУТИ останні стовпці матриці U і останні рядки матриці V^t, залишивши тільки перші 2. Це відповідно координати x, y кожного слова для матриці U і координати x, y для кожного документа в матриці V^t. Розкладання такого виду називають двовимірним сингулярним розкладанням. *ПЕРЕЙТИ НА* Крок 7

Крок 7. ПІДГОТУВАТИ вихідні дані у вигляді вкладених списків координат x, y для двох стовпців матриці U слів і двох рядків матриці V^t документів. *ПЕРЕЙТИ НА* Крок 8

Крок 8. ПОРІВНЯТИ координат слів заданого словника (у випадку задачі перевірки вмотивованості) або термами областей знань (у випадку задачі визначення дисциплін) з відомими документами за допомогою косинусної відстані.

3.4.2. Латентне розміщення Діріхле

Розглянемо схему алгоритму латентного розміщення Діріхле [16].

Крок 0. Наближено ОЦІНИТИ (або ЗАДАТИ) середнє число тем в документі Kі середнє число ключових слів в темі V. *ПЕРЕЙТИ НА* Крок 1

Крок 1. ЗМОДЕЛЮВАТИ розподіл тем по документам і словам за розміщенням Діріхле з параметрами: K-вимірний вектор α (чим менше α, тим менше виражених тим в документі), V-вимірний вектор β (чим менше β, тим менше слів, що характеризують тему). Зазвичай всі координати векторів α і β беруть однаковими. *ПЕРЕЙТИ НА* Крок 2

Крок 2. Для кожного документа ЗГЕНЕРУВАТИ ймовірність тем із розподілу Діріхле з параметром α: *ПЕРЕЙТИ НА* Крок 3

Крок 3. Для кожної теми ЗГЕНЕРУВАТИ ймовірність слів із розподілу Діріхле з параметром α: *ПЕРЕЙТИ НА* Крок 4

Крок 4. Для кожної позиції в документі: ВИБРАТИ випадково тему згідно згенерованим ймовірностям, ВИБРАТИ випадково слово, згідно ймовірностям слів в темі.

Висновок до розділу

У розділі математичного забезпечення була сформульована змістовна та математична постановки задачі формування індивідуальної освітньої траєкторії.

Були також сформульовані допоміжні задачі визначення вмотивованості студентів та задача визначення дисциплін, яка була розбита на підзадачі визначення списку областей знань для професії та визначення ключових слів

з тематик обраних областей знань. Виконано зведення допоміжних задач до задач штучного інтелекту (методи латентно-семантичного аналізу та латентного розміщення Діріхле).

Для вирішення задачі перевірки вмотивованості було обрано метод латентно-семантичного аналізу, а для реалізації класифікатора – схема TF-IDF. Для задачі визначення дисциплін було також обрано метод латентно-семантичного аналізу (підзадача визначення списку областей знань для професії) та метод латентного аналізу Діріхле (підзадача визначення ключових слів із тематик обраних областей знань) Ці два методи відомі дуже високою якістю рішень, саме тому було вирішено їх обрати

Також наведено детальний опис та схеми алгоритмів семантичного аналізу тексту та латентного розміщення Діріхле.

4 ПРОГРАМНЕ ТА ТЕХНІЧНЕ ЗАБЕЗПЕЧЕННЯ

4.1 Засоби розробки

При створенні програмного продукту були використані такі засоби для програмування на Python як Visual Studio IDE з плагіном Python Tools for Visual Studio, а також база даних SQLite.

Мова програмування Python має ряд переваг: дозволяє розробляти веб-застосунки, дозволяє працювати з такими сферами, як аналіз даних і машинне навчання (пакети scipy, scikit-learn, pandas, numpy), має велику кількість готових бібліотек, низький поріг входження в мову та високу продуктивність програмістів, які пишуть на Python [18].

Мова Python дуже проста. Простота досягається за рахунок того, що мова і всі її складові від самого початку є такими, що інтерпретуються.

Наступна важлива перевага – універсальність і «всеїдність».

По-перше, програми на Python можуть працювати з будь-якою операційною системою. Це робить її універсальною, як для користувацьких задач, застосунків у гаджетах, системних задач на серверах і масивах даних і т.д. Багато в чому завдяки цьому Python і набрала свою популярність. Більш того, на відміну від свого попередника C, Python займає рівну кількість ресурсів пам’яті, а тому є по-справжньому легко переносним кодом.

По-друге, мова може виконувати практично будь-яке завдання за рахунок легкого розширення певними модулями. До основної програми завжди можна підключити специфічні функції, реалізовані в мові C. Використовуючи такі готові конструкції, можна вирішувати складні завдання, пов’язані з обробкою графіки, математичними обчисленнями, візуальним моделюванням. Крім того, всередині самого Python кожен користувач має можливість створювати універсальні «добавки», доступні абсолютно всім. Відкритість цієї мови програмування – саме те, заради чого все і створювалося першопрохідцями Python. Завдяки безкоштовності платформ і модулів Python

відмінно підходить як професіоналам, так і любителям. Розробнику не доведеться витрачатися ні на які ліцензії та модулі, все є у відкритому доступі.

Виділимо деякі недоліки мови Python. Загалом недоліків у цієї мови зовсім небагато. Більшість програмістів погоджуються, що все ж Python не такий швидкий, як хотілося б. Навіть у порівнянні з іншими мовами, він може програвати. Проте для більшості комп'ютерів і тим більше гаджетів жвавості Python більш ніж достатньо.

Програмісту, що працює саме з цією мовою, доведеться враховувати, що для гідної роботи необхідно створювати максимально логічний і простий код. Особливо це стосується створення кодів для серверів і інтернет-ресурсів. Щоб писати саме швидкі коди на Python потрібна тривала практика і нестандартне мислення поза рамками готових модулів.

Python має ряд особливостей, які не можуть бути зараховані ні до переваг, ні до недоліків.

Одна з особливостей – відсутність змінних, точніше, необхідність їх прописувати. Для тих, хто довго працював з іншими мовами, це може бути складністю при переході на Python. Однак для новачків і тих, хто переходить на Python після інших мов, це навпаки буде суттєвою перевагою.

З'являються в Python і своєрідні терміни, пов'язані в першу чергу, з його можливостями. Якщо для багатьох мов робота з великими списками представляла складність, то в Python такі функції реалізуються через особливі об'єкти: тьюпли, словники і карти. До них теж доведеться звикнути. Невеликі труднощі у новачків може викликати об'єктно-орієнтованість, але при хорошому рівні навчання це також можна перетворити в гідність мови [19].

Python є однією із найбільш затребуваних мов, використовується в цілях створення програм і додатків в наступних областях:

– в машинобудуванні, приладобудуванні, медицині, бізнесі, менеджменті, силовий електроніці, банківських системах, в комерційних

структурах, в області управління інформаційних комунікацій, для забезпечення функціонування систем безпеки підприємств, різних галузях промисловості, картографії та геодезії, дизайні, медіаіндустрії, засобах масової інформації, в сфері розробки біотехнологій, в сфері сервісу, у фізиці і механіці тощо;

– розробка засобів реалізації програм (методичних, інформаційних, технічних, програмних, математичних);

– з метою налаштування технічних засобів, які забезпечують введення систем в експлуатацію [20].

SQLite – база даних, яка легко вбудовується в додатки. Через те, що ця система керування базами даних базується на файлах, то вона надає достатньо великий набір інструментів для роботи з нею. При роботі з цією СКБД звертання відбуваються безпосередньо напряму до файлів (в яких зберігаються дані), замість портів в мережевих СКБД. З цих причин СКБД SQLite дуже швидка та могутня враховуючи технології обслуговуючих бібліотек.

Наведемо основні переваги SQLite:

– структура файлів – уся база даних (сховище інформації) складається з одного файлу, тому її надзвичайно легко переносити на різні комп’ютери;

– стандарти – ця СКБД використовує SQL. Деякі речі у ній все ж упущені (наприклад, RIGHT OUTER JOIN або FOR EACH STATEMENT), але основні все-таки підтримуються;

– зручна при тестуванні та розробці – у процесі розробки застосувань часто виникає необхідність виконання масштабування. SQLite складається всього з одного файлу і бібліотеки написаною мовою програмування С, таким чином пропонучи все для досягнення цілей масштабування.

Наведемо також недоліки SQLite:

– відсутність системи користувачів – великі СКБД включають в свій склад системи, котрі допомагають управляти правами доступу користувачів. Звичайно, застосування цієї функції не таке критичне, адже ця СКБД використовується в невеликих додатках;

– відсутність можливості збільшення продуктивності.

Таким чином, SQLite зручно використовувати для вбудованих застосувань, якщо важлива можливість легкого перенесення програми, але не важлива масштабованість [21].

4.2 Вимоги до технічного забезпечення

4.2.1 Загальні вимоги

Розроблений програмний продукт представляє собою комплекс основних функцій процесу формування індивідуальної освітньої траєкторії і призначений для використання майбутніми студентами, стейкхолдерами та працівниками вищих навчальних закладів.

Технічне забезпечення системи повинно максимально і найбільш ефективним чином використовувати існуючі технічні засоби.

Для коректної роботи розробленої програми до складу технічних засобів повинні входити:

– комп’ютер з такою конфігурацією:

1) процесор з тактовою частотою не нижче 1.6 ГГц;
2) достатній об’єм оперативної пам’яті (2 Гб і вище);
3) інші складові технічних засобів можуть мати будь-які параметри, через те, що вони не впливають на роботу програми.

– додатково має бути встановлене наступне програмне забезпечення для програмної та інформаційної сумісності:

1) операційна система Windows 7 та вище;
2) середовище розробки Visual Studio 2017 та вище;

3) мова програмування Python 3.6 і вище;
4) система керування базою даних SQLite;
5) Net Framework 3.5 і вище;
6) Microsoft Word 2007 і вище.

– комп'ютерна периферія, до складу якої входить:

1) монітор;
2) клавіатура;
3) мишка;
4) принтер.

Для використання системи на мобільній платформі, пристрій повинен працювати на операційній системі Android версії не нижче 4.0 або IOS не нижче 10 та мати можливість виходу в Інтернет.

4.3 Архітектура програмного забезпечення

Архітектура застосвування побудована за патерном Model View Controller (MVC). Використаний фреймворк Django слідує концепції MVC досить строго. Проте найбільше дій відбувається в моделях, шаблонах і уявленнях, Саме тому Django може називатися «MTV-фреймворком».

У концепції MTV:

– M означає «Model» – модель, рівень доступу до даних;

– T означає «Template» – шаблони. Цей рівень містить в собі все, що пов'язано з представленням: як саме щось повинно бути показано на веб-сторінці або будь-якому іншому документі;

– V означає «View» – представлення. Цей рівень пов'язаний з усією логікою, що зв'язує модель і необхідні шаблони [22]. На рисунку 4.1 зображено архітектуру django-застосування [23].

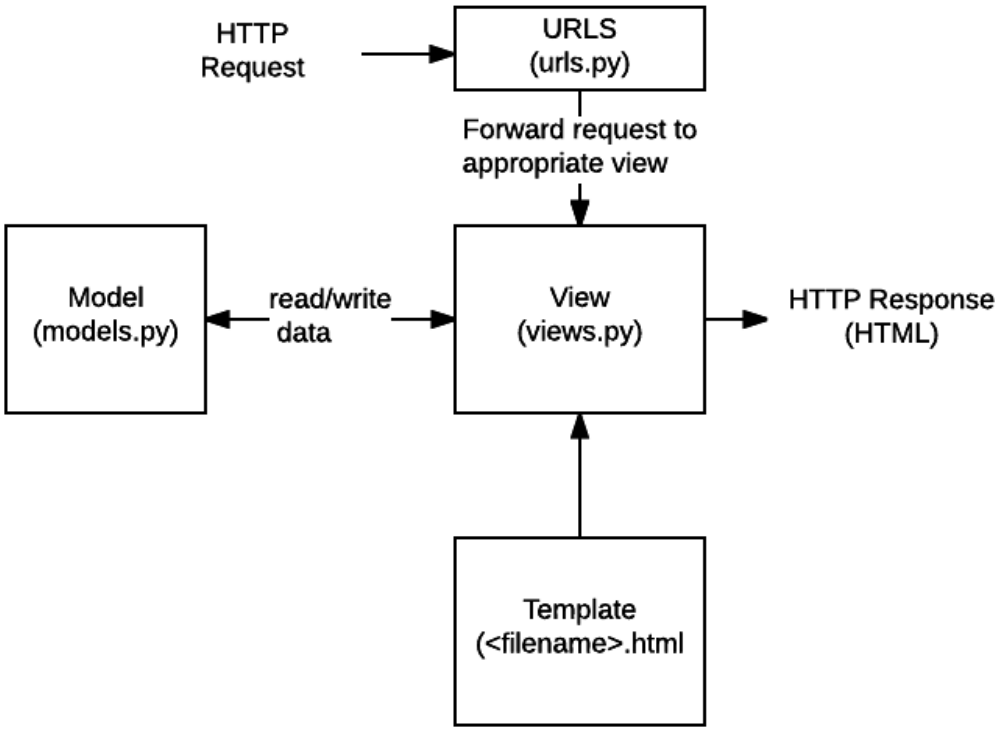


Рисунок 4.1 – Архітектура застосування

Доцільна побудова наступних діаграм Unified Modeling Language (UML) для демонстрації архітектури програмного забезпечення: класів, послідовності, компонентів.

№	Тип діаграми	Роль у процесі проектування
1	Діаграма класів	Визначає набір класів та їхню взаємодію задля вирішення задачі побудови індивідуальної освітньої траєкторії.
2	Діаграма послідовності	Дана діаграма містить набір класів та їхню послідовність взаємодії для вирішення задачі побудови індивідуальної освітньої траєкторії.
3	Діаграма компонентів	Визначає групування класів у компоненти та ланцюжки виклику компонентів іншими компонентами під час виконання програми.

4.3.1 Діаграма класів

Структурна схема класів наведена у графічному матеріалі. Класи відповідають за виконання таких функцій програми, як формування списку компетентностей та дисциплін претенденту відповідно до результатів тестів, а також формування навчального плану.

Опишемо основні класи діаграми:

- «User» (Користувач) – клас програми, який визначає користувачів системи, містить дані про них: ім’я, прізвище, унікальне ім’я користувача (username), e-mail, пароль та роль. Користувачі мають можливість реєструватися, а також входити та виходити з системи;
- «Applicant» (Претендент) – клас, що відповідає за претендентів системи, визначає результати їх тестів, наявність мотивації та певні документи, що необхідні для формування індивідуальної траєкторії, також дані про те, чи були сформовані дисципліни і чи були обрані стейкхоледром дисципліни затверджені. Претендент має можливість пройти тести, виявити мотивацію, сформувати траєкторію та обрати дисципліни;
- «Stakeholder» (Стейкхолдер) – клас, у якому містить інформація про стейкхолдера: назва та вид діяльності компанії, в якій він працює, а також його посада. Повинен затвердити обрані претендентом навчальні дисципліни;
- «Tutor» (Тьютор) – клас, у якому містить інформація про тьютора: вид його діяльності, повинен визначити дисципліни по ключовим словам для певного претендента;
- «Job» (Професія)– клас, який містить загальну інформацію про усі можливі професії IT-сфери: назва;
- «Competence» (Компетентність) – клас, який містить загальну інформацію про усі компетентності відповідно до європейської рамки компетентностей: назва, знання та вміння;

– «Knowledge Area» (Область знань) – клас, який містить загальну інформацію про усі області знань: назва, її тематики;

– «Discipline» (Дисципліна) – клас, який містить загальну інформацію про сформовані тьюторами дисципліни: назва дисципліни, кількість кредитів та форма конролю;

– «Curriculum» (Навчальний план) – клас, який містить дані про сформовану індивідуальну освітню траєкторію для претендента: назва, галузь знань;

– «LSA» (Латенто-семантичний аналіз) – клас, який містить дані про математичні алгоритми системи, необхідний для аналізу тексту.

Інші класи необхідні для авторизації в системі, а також для використання сесій: LogEntry, Group, Permission, ContentType, PermisionsMixin, AbstractBaseUser, Session, AbstractBaseSession.

4.3.2 Діаграма послідовності

Діаграма послідовності для процесу формування індивідуальної освітньої траєкторії наведена у графічному матеріалі.

У таблиці 4.1 показано, які елементи задіяні на діаграмі.

Таблиця 4.1 – Елементи, які задіяні на діаграмі

Елемент	Відповідальність
Актор «Претендент»	Реєстрація, авторизація, проходження тестів, виявлення мотивації, запит на визначення дисциплін тьютором, перегляд дисциплін та їх вибір, відправлення дисциплін на затвердження стейкхолдеру, перегляд сформованої траєкторії
Актор «Тьютор»	Реєстрація, авторизація, визначення дисциплін для претендента по ключовим словам областей знань
Актор «Стейкхолдер»	Реєстрація, авторизація, затвердження обраних претендентом дисциплін

Продовження таблиці 4.1

Елемент	Відповідальність
Сутність «Професія»	Визначає професії відповідно до європейської рамки компетентностей
Сутність «Компетентність»	Визначає компетентності для професій відповідно до європейської рамки компетентностей
Сутність «Область знань»	Визначає область знань інформатики зі своїми тематиками відповідно до асоціації обчислюваної техніки
Сутність «Дисципліни»	Визначає дисципліни, які були сформовані тьютором на вимогу певного претендента
Сутність «Індивідуальна освітня траєкторія»	Визначає набір дисциплін, які були обрані претендентом та затверджені стейкхолдером
Об’єкт «БД»	БД – база даних, яка надає користувачам доступ до таблиць. Збережені процедури на сервері БД, працюючи безпосередньо з таблицями, серед інших виконують такі функції: – збереження даних; – виконує запити на виявлення певних даних

4.3.3 Діаграма компонентів

На рисунку 4.2 наведемо діаграму компонентів системи.

У таблиці 4.2 опишемо модулі та інтерфейси діаграми:

Таблиця 4.2 – Модулі та інтерфейси діаграми

Модуль	**Інтерфейс, що надається**	**Необхідний інтерфейс**	**Опис**
Модуль претендентів	Результати тестів, мотивація, особисті дані, обрані дисципліни	Список професій, список дисциплін, готовий план	Містить у собі усіх претендентів системи
Модуль стейкхолдерів	Результат затвердження	Обрані дисципліни	Містить у собі усіх стейкхолдерів системи

Продовження таблиці 4.2

Модуль	**Інтерфейс, що надається**	**Необхідний інтерфейс**	**Опис**
Модуль тьюторів	Сформовані дисципліни	Особисті дані, ключові слова областей знань	Містить у собі усіх тьюторів системи.
Модуль професій	Ключові слова областей знань, список професій	Відповідні компетентності, інформація про професії, відповідні області знань	Містить у собі набір професій системи
Модуль дисциплін	Інформація про сформовані дисципліни, список дисциплін	Сформовані дисципліни, інформація про дисципліни	Містить у собі набір дисциплін системи, сформованих тьютором
Модуль компетентностей	Відповідні компетентності	Інформація про компетентності	Містить у собі набір компетентностей до професій
Модуль областей знань	Відповідні області знань	Інформація про області знань	Містить у собі набір областей знань інформатики
Модуль навчальних планів	Інформація про сформований навчальний план, готовий план	Інформація про навчальний план, результат затвердження	Містить у собі створені траєкторії
База даних	Інформація про дисципліни, інформація про професії, інформація про компетентності, інформація про області знань, інформація про навчальний план	Результати тестів, мотивація, інформація про сформовані дисципліни, інформація про сформований навчальний план	База даних системи

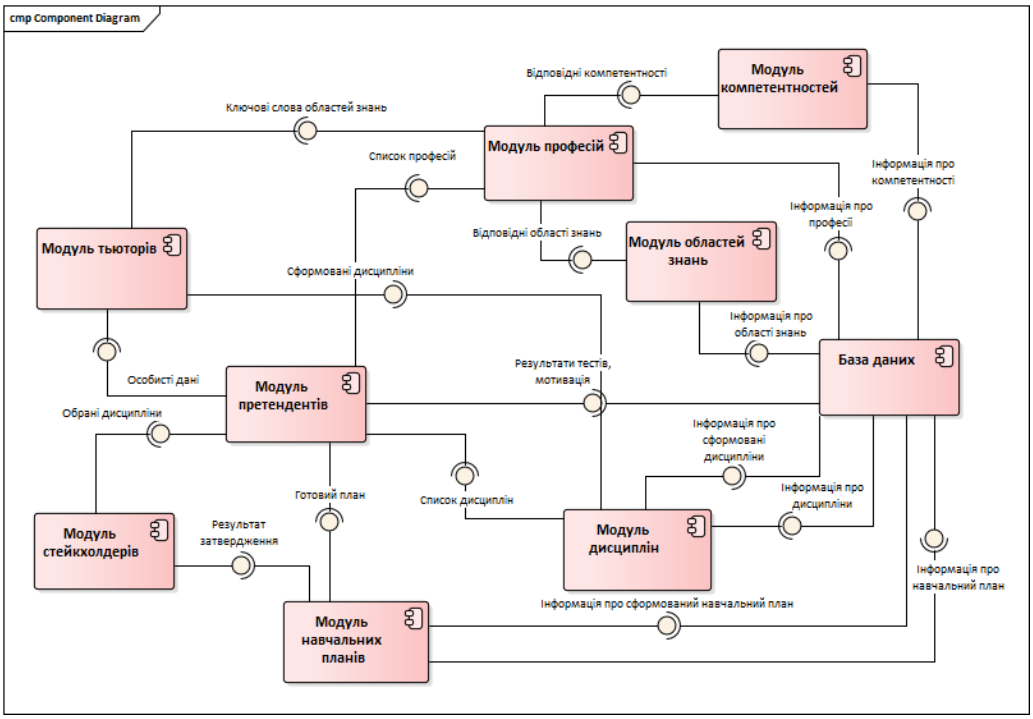


Рисунок 4.2 – Схема структурна компонентів системи побудови персональних освітніх траєкторій

4.3.4 Специфікація функцій

Функції класів програмного забезпечення наведені в таблиці 4.3.

Таблиця 4.3 – Функції класів програмного забезпечення

Назва	Примітка
Клас: Користувач –клас, що відповідає за користувачів системи, містить їх особисті дані та є батьківським класом для таких класів: Претендент, Стейкхолдер, Тьютор.	
+ Login(): void	Виконує вхід користувача у систему
+ Log off(): void	Реалізує вихід користувача із системи
+ Register(): void	Виконує реєстрацію користувача в системі
Клас: Претендент –клас, що відповідає за претендентів системи, містить дані, необхідні для побудови індивідуальної освітньої траєкторії.	
+ PassPsychologicalTest(): void	Реалізує аналіз пройденого претендентом психологічного тесту та виконує запис результату тесту до бази даних

Продовження таблиці 4.3

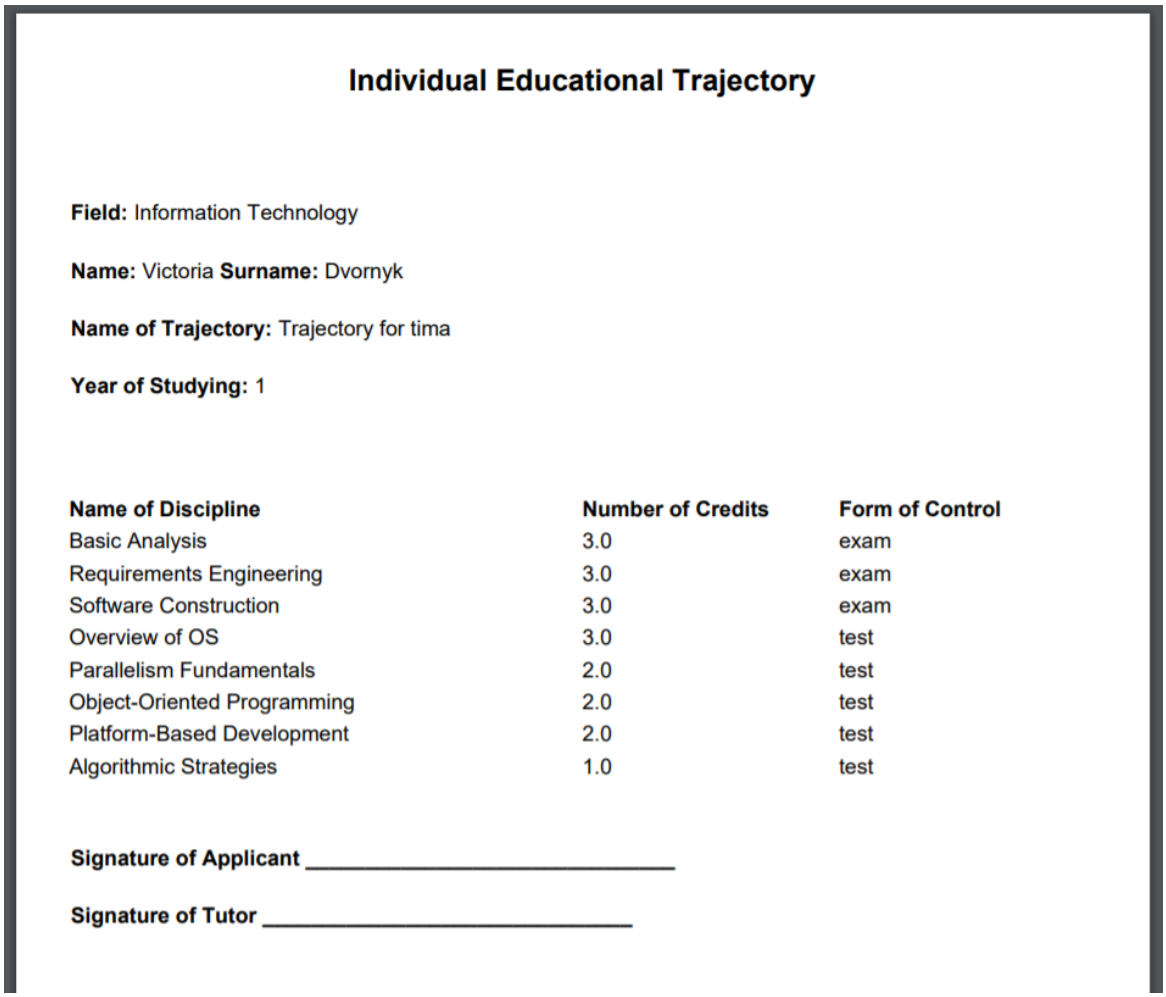
Назва	Примітка
+ PassLidershipTest(): void	Реалізує аналіз пройденого претендентом тесту на визначення лідерських якостей та виконує запис результату тесту до бази даних
+ PassCompetence &ProficiencyTest(): void	Реалізує аналіз пройденого претендентом тесту на професійні компетентності та виконує запис результату тесту до бази даних
+ FindOutJob(): void	Виконує запис обраної професії до бази даних
+ FindOutMotivation(): void	Виконує визначення мотивації у претендента та запис результату до бази даних
+ MakeTrajectory(): void	Виконує відправлення запиту на формування тьютором дисциплін
+ ChooseDisciplines (): void	Виконує відправлення обраних дисциплін стейкхолдеру для їх затвердження
+ ReviewOfTrajectory (): void	Застосовується для перегляду сформованої траєкторії
Клас: Стейкхолдер –клас, що відповідає за стейкхолдерів системи, містить дані, необхідні ідентифікації напряму роботи стейкхолдера.	
+ ApproveDisciplines(): void	Застосовується для затвердження обраних претендентом дисциплін
Клас: Тьютор –клас, що відповідає за тьюторів системи, містить дані, необхідні ідентифікації сфери діяльності тьютора.	
+ DetermineDisciplines(): void	Застосовується для визначення дисциплін претенденту

4.4 Опис звітів

За допомогою програми можлива побудова індивідуальної освітньої траєкторії.

Індивідуальна освітня траєкторія генерується після затвердження стрейкхолдером обраних претендентом дисциплін і з’являється у пункті меню: «YOUR TRAJECTORY» на сторінці претендента автоматично. При виборі в

програмі цього пункту меню можна завантажити готову траєкторію у форматі PDF. Згенерована індивідуальна траєкторія має вигляд зображений на рисунку 4.3, де показано назву галузі знань, дані претендента (прізвище та ім'я), назва траєкторії, номер курсу (рік навчання), на який була сформована траєкторія, а також список дисциплін, які претендент буде вивчати, і загальна інформація про них (кількість кредитів та форма контролю).



Individual Educational Trajectory

Field: Information Technology

Name: Victoria **Surname:** Dvornyk

Name of Trajectory: Trajectory for tima

Year of Studying: 1

Name of Discipline	Number of Credits	Form of Control
Basic Analysis	3.0	exam
Requirements Engineering	3.0	exam
Software Construction	3.0	exam
Overview of OS	3.0	test
Parallelism Fundamentals	2.0	test
Object-Oriented Programming	2.0	test
Platform-Based Development	2.0	test
Algorithmic Strategies	1.0	test

Signature of Applicant _____

Signature of Tutor _____

Рисунок 4.3 – Вигляд індивідуальної освітньої траєкторії

Висновок до розділу

У даному розділі було визначено програмне забезпечення для розробки програмного продукту. Було використано такі засоби для програмування на Python як Visual Studio IDE з плагіном Python Tools for Visual Studio, а також

база даних SQLite. Також визначено переваги та недоліки засобів розробки, і наведено загальні вимоги до технічного забезпечення.

Для формування архітектури програмного забезпечення було побудовано ряд діаграм: класів, послідовності та компонентів. Також наведено опис цих діаграм.

Діаграма послідовності визначила у первинному наближенні множини та інтерфейси класів, а в діаграмі класів класи було визначено остаточно.

Для збереження множини і атрибутів класів в базі даних передбачаються таблиці. Тому було проведено певну відповідність між множиною класів та множинами сутностей на ER-моделі структури БД, а також на відповідність зв’язків асоціації між класами та між належними множинами сутностей на ER-діаграмі.

Діаграма компонентів визначила групування класів у компоненти.

Також було описано індивідуальну освітню траєкторію, яка буде сформована у результаті роботи програми, та показано її вигляд.

5 ТЕХНОЛОГІЧНИЙ РОЗДІЛ

5.1 Керівництво користувача

Для запуску програмного застосування потрібно відкрити файл з розширенням .sln у Visual Studio 2017 IDE, після цього, для запуску серверу натиснути кнопку виділену на рисунку 5.1.

Рисунок 5.1 – Кнопка запуску сервера

Користувач побачить головне вікно застосування, яке містить навігаційну панель та загальну інформацію про застосування. Головну сторінку застосування показано на рисунку 5.2.

Рисунок 5.2 – Головна сторінка програмного застосування

Користувач має змогу авторизуватись у системі. Для авторизації переходимо на сторінку авторизації, зображену на рисунку 5.3.

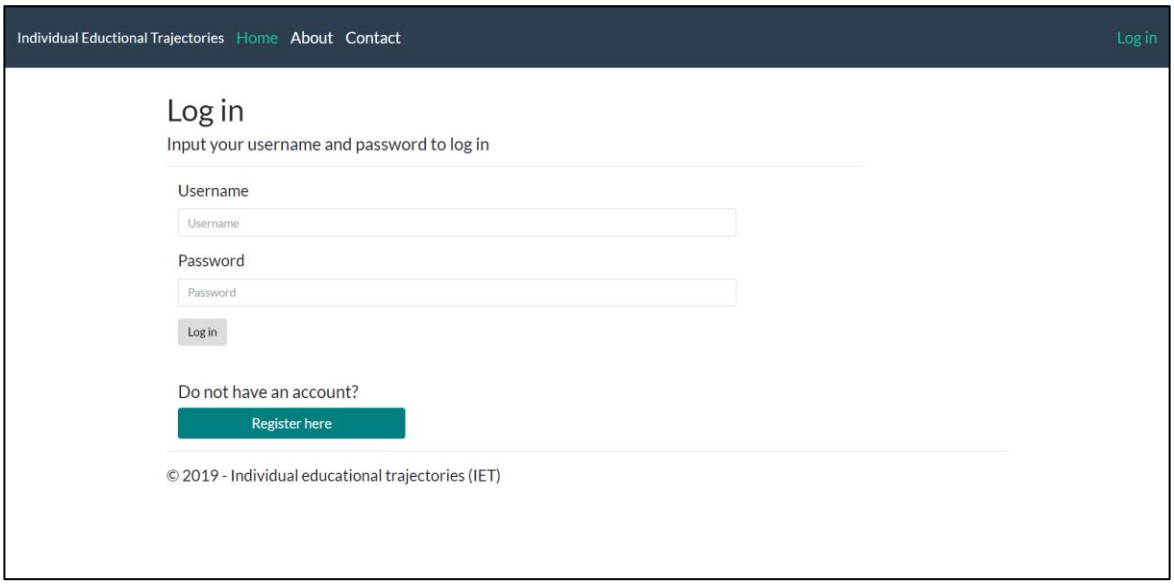


Рисунок 5.3 – Сторінка авторизації

На цій сторінці користувач повинен ввести свої особисті дані (username та пароль) для того, щоб увійти в систему. Для процесу авторизації присутні такі перевірки:

– на заповненість полів;

– на присутність користувача з введеним унікальним іменем та паролем у базі даних.

Якщо дані було введено і помилок не виникло, користувач потрапляє на головну сторінку (рисунок 5.2), інакше, отримує повідомлення про те, що його не вдалося ідентифікувати у базі даних, тобто введено невірний username та/або пароль. Користувачеві пропонується повторно здійснити вхід у систему (кількість таких спроб не обмежена). Приклад авторизації з помилками зображено на рисунку 5.4.

Рисунок 5.4 – Невдала авторизація користувача

Як видно з рисунку 5.4 користувач спробував ввести неіснуючий username та пароль, після чого отримав повідомлення «Please enter a correct username and password» (Будь ласка, введіть коректний username та пароль).

Якщо у користувача відсутній акаунт, він може здійснити реєстрацію в системі. Сторінка реєстрації зображена на рисунку 5.5. У системі є можливість реєстрація трьох типів: претендентів, стейкхолдерів та тьюторів. На рисунку 5.5 зображена саме реєстрація тьютора. Кожному користувачеві пропонується ввести своє ім’я, прізвище, електронну пошту, пароль та вказати свою роль. Стейкхолдерам потрібно ввести дані про свою компанію: її назву, вид діяльності та свою посаду. Тьюторам порібно додатково ввести вид своєї діяльності. Для процесу реєстрації присутні такі перевірки:

– на заповненість полів;

– на коректність введення електронної пошти (пошта повинна містити обов’язково знак @, наприклад, tutor@tutor.com);

– на унікальність поля «Username».

Рисунок 5.5 – Сторінка реєстрації

Якщо дані було введено і помилок не виникло, користувач потрапляє на головну сторінку (рисунок 5.2), інакше, отримує повідомлення про те, що його не вдалося зареєструвати, тобто введено невірні дані. Користувачеві пропонується повторно здійснити реєстрацію (кількість таких спроб не обмежена). Приклад реєстрації з помилками зображено на рисунку 5.6.

Рисунок 5.6 – Невдала реєстрація користувача

Як видно з рисунку 5.6 користувач спробував ввести існуючий username, після чого отримав повідомлення «This username already exists» (Користувач з таким username вже існує).

Авторизований користувач може переглянути свою сторінку, вона має вигляд сторінки, зображеної на рисунку 5.7. На сторінці міститься інформація про користувача: його ім’я, прізвище, username та електоронна пошта. Також є такі кнопки:

– «MAKE TRAJECTORY» – здійснює перехід на сторінку формування траєкторії (рисунок 5.8);

– «CHOOSE DISCIPLINES» – здійснює перехід на сторінку вибору дисциплін для користувача;

– «YOUR TRAJECTORY» – здійснює перехід на сторінку перегляду створеної траєкторії.

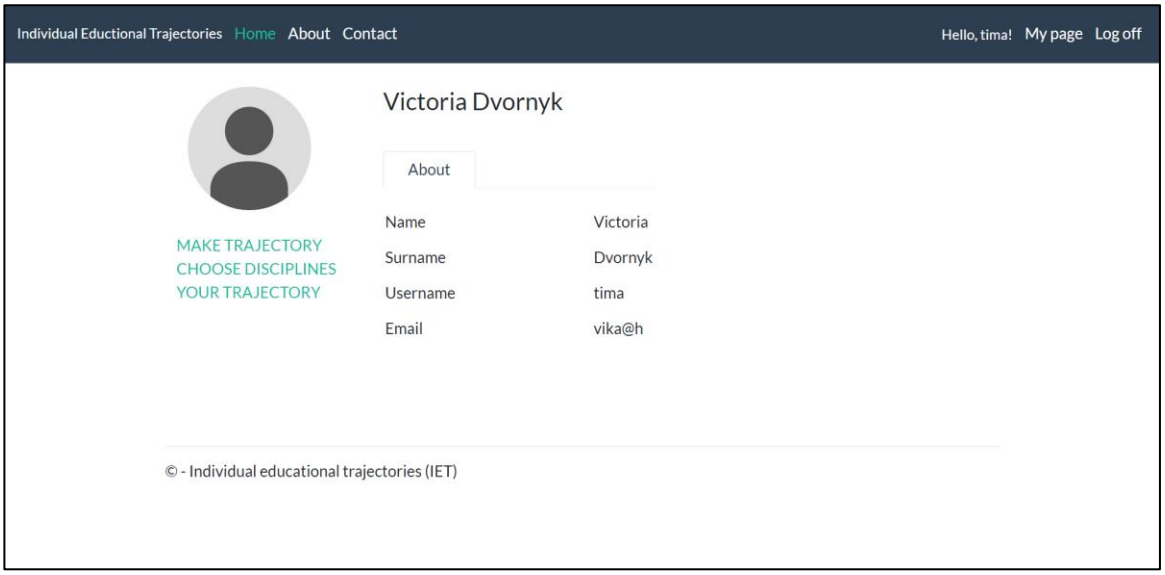


Рисунок 5.7 – Сторінка претендента

Для того, щоб створити траєкторію, перейдемо на сторінку створення траєкторії (рисунок 5.8), натиснувши кнопку «MAKE TRAJECTORY» на сторінці користувача. На рисунку 5.8 бачимо вже заповнений профіль користувача. Для початку користувач пройшов психологічний тест [24], далі тест на лідерство [25], після цього тест на професійні компетентності [26].

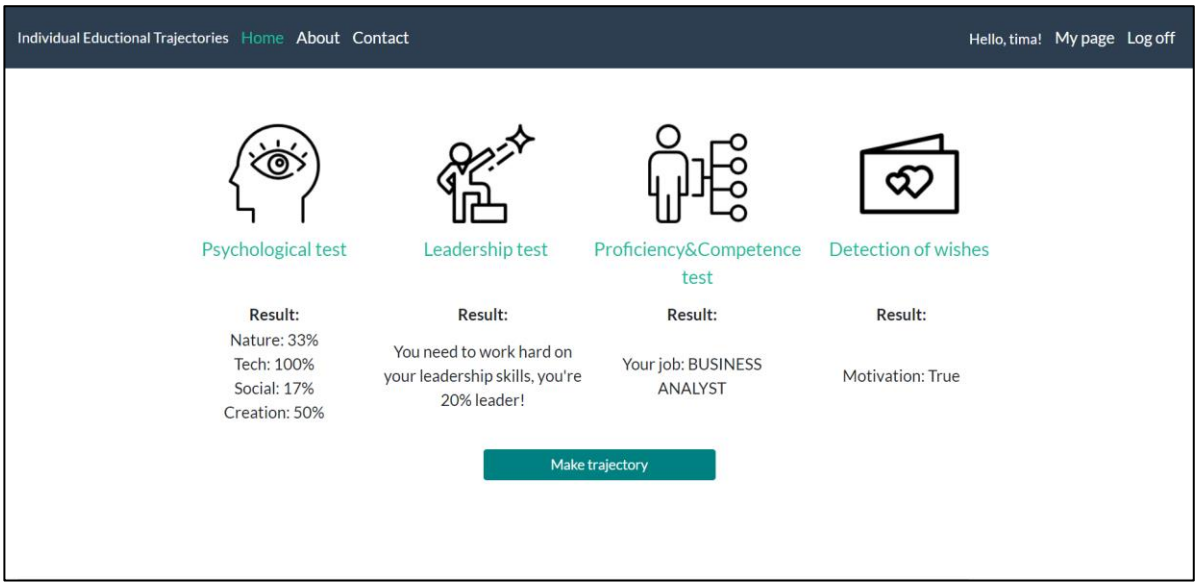


Рисунок 5.8 – Сторінка створення індивідуальної траєкторії

Після проходження тесту на професійні компетентності, користувач повинен обрати професію, котру він отримав в результаті тестування. Сторінка вибору професії зображена на рисунку 5.9.

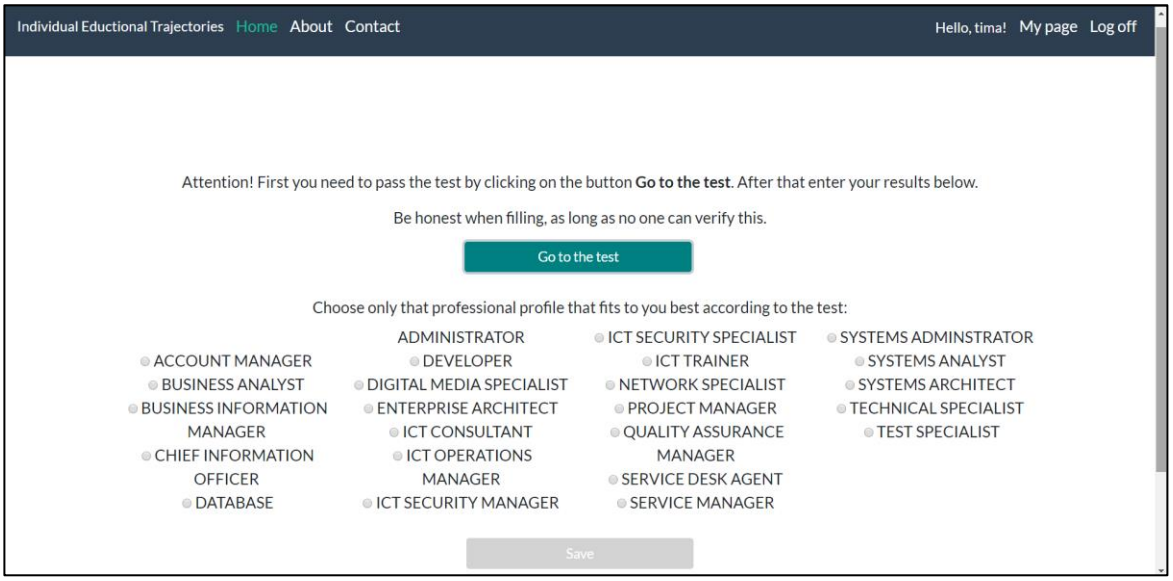


Рисунок 5.9 – Сторінка вибору професії

Після вибору професії користувачеві доступна сторінка визначення його мотивації (рисунок 5.10). Претендентові пропонується вставити у поле його мотиваційний лист або есе. Після чого система визначить, чи є мотивація у користувача.

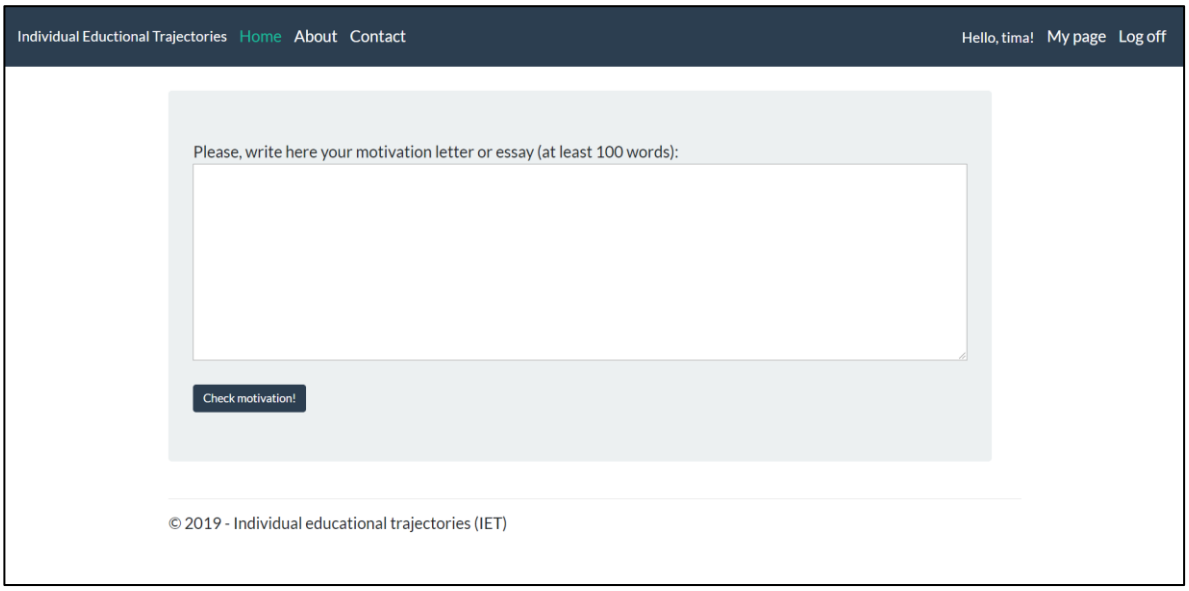


Рисунок 5.10 – Сторінка визначення мотивації претендента

Далі повернувшись на сторінку формування індивідуальної траєкторії (рисунок 5.8) користувачеві потрібно натиснути на кнопку «Make trajectory». Користувач побачить повідомлення про те, що його дані були відправлені тьютору для формування дисциплін.

Коли тьютор зайде до себе в кабінет (рисунок 5.11), він може переглянути запити на формування дисциплін (рисунок 5.12).

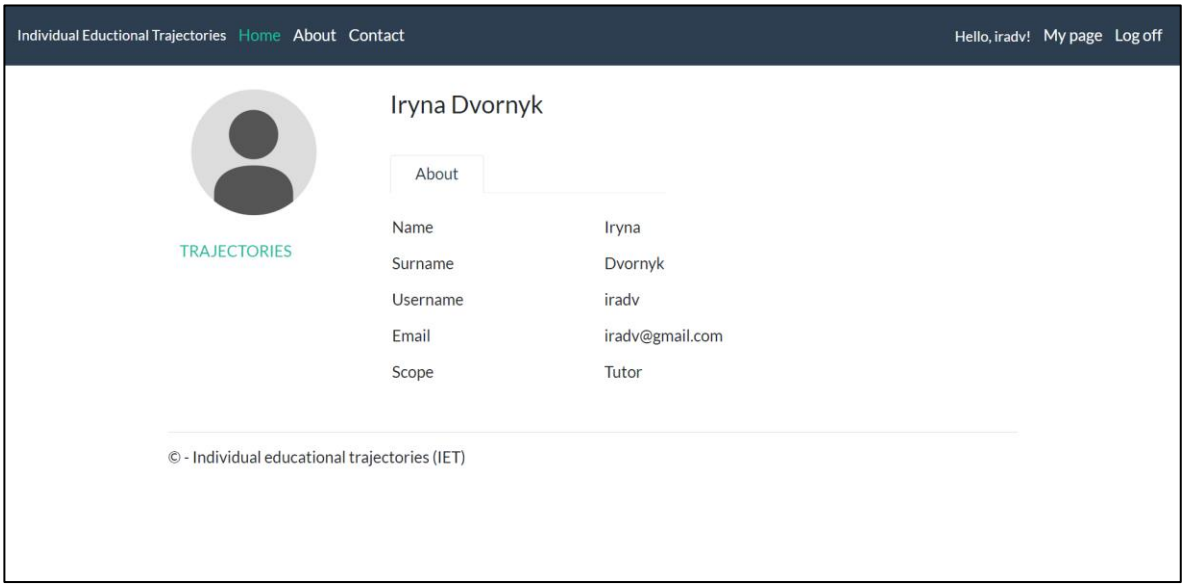


Рисунок 5.11 – Сторінка тьютора

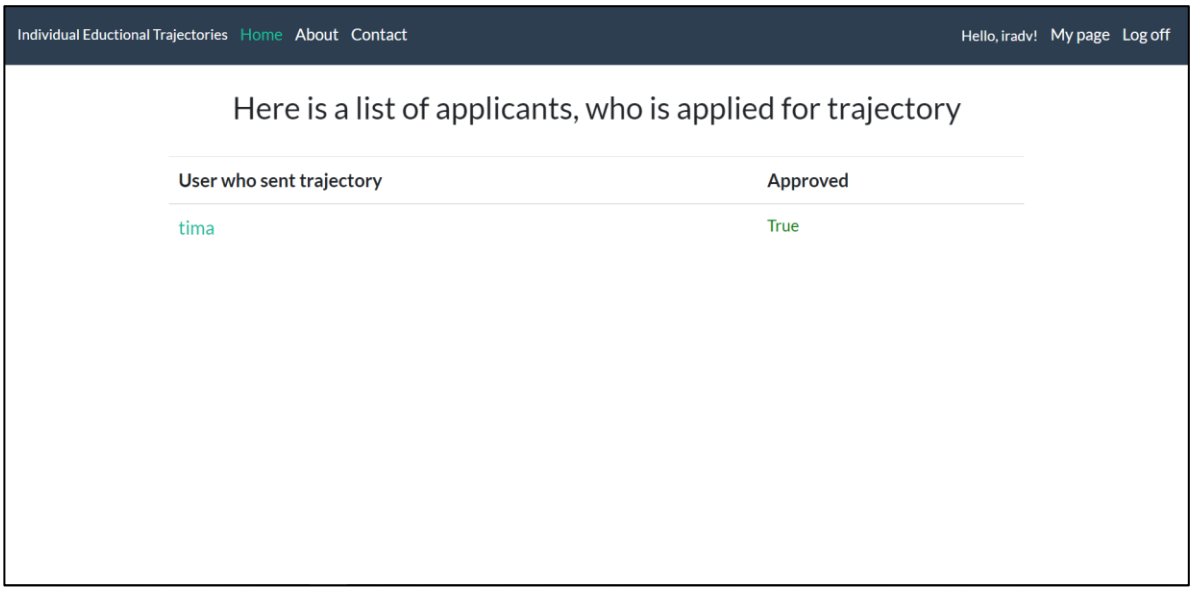


Рисунок 5.12 – Сторінка визначення мотивації претендента

На рисунку 5.12 показано користувача, а також те, чи було вже сформовано йому дисципліни. Значення «True» у колонці «Approved» означає, що цьому користувачеві дисципліни вже було сформовано тьютором. Якщо тьютор прагне визначити дисципліни для користувача, він потрапляє на сторінку визначення дисциплін (рисунок 5.13).

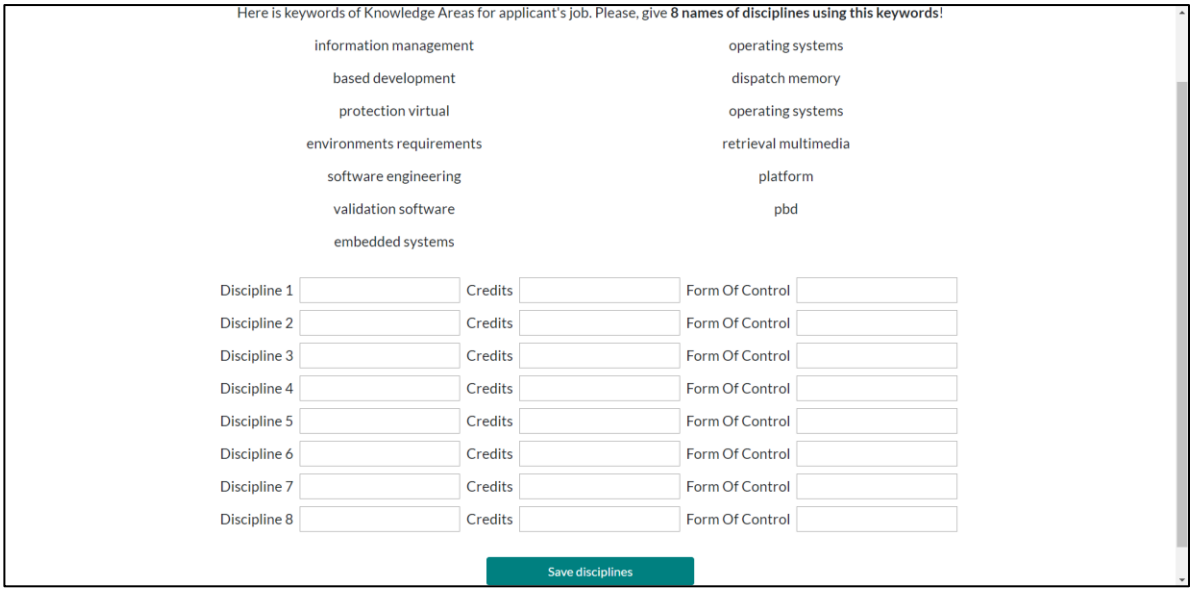


Рисунок 5.13 – Сторінка визначення дисциплін тьютором

На сторінці визначення дисциплін тьютор переглядає ключові слова областей знань для професії, яку обрав претендент. Далі йому обов'язково потрібно визначити назви (по заданим ключовим словам), кількість кредитів

та форму контролю рівно вісьмом дисциплінам. Після заповнення даних про дисципліни потрібно натиснути кнопку «Save disciplines», і дисципліни буде збережено до бази даних.

Після збереження дисциплін користувач вже може обрати дисципліни у себе в кабінеті, натиснувши на кнопку «CHOOSE DISCIPLINES». Вікно вибору дисциплін зображено на рисунку 5.14. Претенденту показано усі можливі дисципліни, котрі були сформовані різними тьюторами для обраної претендентом професії. Йому пропонується обрати рівно 8 дисциплін для вивчення.

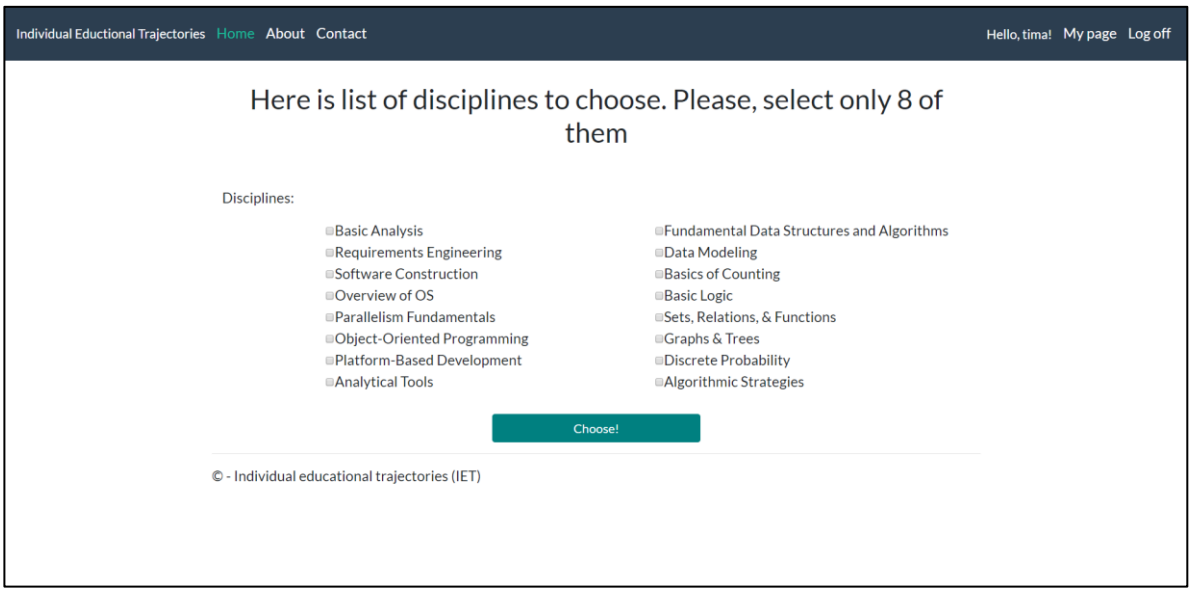


Рисунок 5.14 – Сторінка вибору дисциплін претендентом

Після вибору дисциплін, претендент натискає на кнопку «Choose». Обрані дисципліни записуються до бази даних, та надсилаються на перевірку стейкхоледеру. Вікно усіх претендентів, що очікують затвердження зображено на рисунку 5.15.

Рисунок 5.15 – Сторінка списку користувачів, котрі очікують затвердження обраних дисциплін

Для того, щоб затвердити обрані дисципліни, достатньо натиснути на користувача – відбудеться перехід на сторінку затвердження (рисунок 5.16).

Рисунок 5.16 – Сторінка затвердження дисциплін

Після затвердження дисциплін претендент може переглянути створену траєкторію, натиснувши кнопку «YOUR TRAJECTORY» в особистому кабінеті. Приклад створеної траєкторії можна побачити на рисунку 5.17. Якщо стейкхолдер не затвердив траєкторію, претендент може обрати дисципліни повторно, а потім знову надіслати їх стейкхолдеру.

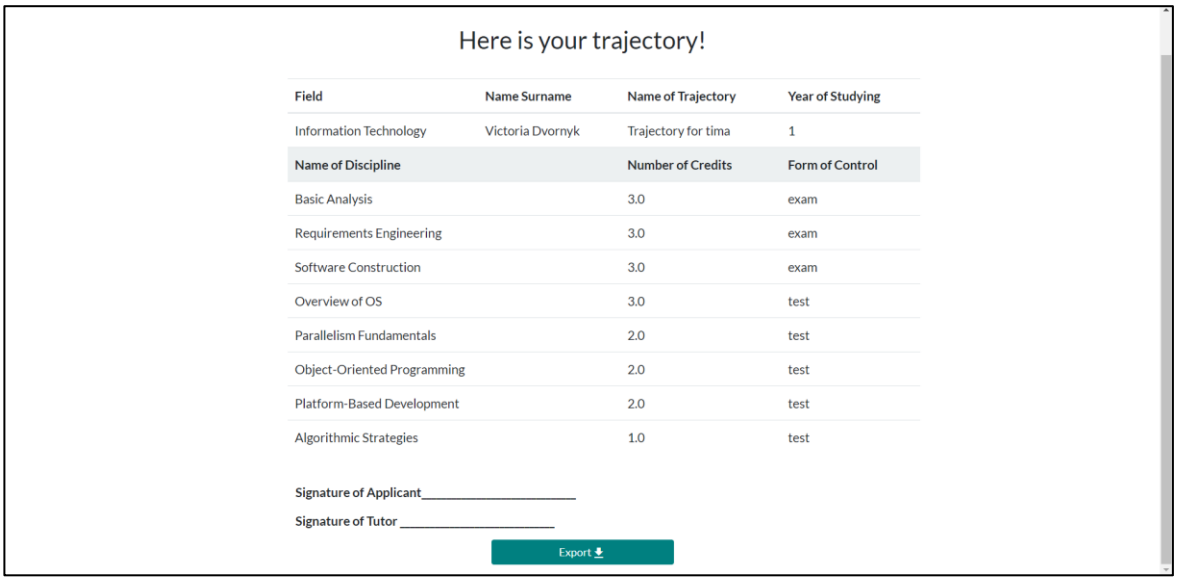


Рисунок 5.17 – Сторінка створеної траєкторії

Траєкторія містить галузь знань, для якої вона була побудована, також містить дані про претендента (прізвище та ім’я), особисту назву траєкторії, номер курсу претендента(рік навчання), на який була сформована траєкторія, а також список дисциплін, які претендент буде вивчати, і загальна інформація про них (кількість кредитів та форма контролю).

Також у претендента є можливість екпортувати створену траєкторію у форматі PDF. Звіт було представлено на рисунку 4.3.

5.2 Випробування програмного продукту

У цьому підрозділі наведено опис тестів і порядок їх виконання для перевірки відповідності програмного забезпечення системи функціональним вимогам, представленим у технічному завданні на створення системи побудови індивідуальних освітніх траєкторій.

5.2.1 Мета випробувань

Метою випробувань є перевірка відповідності функцій системи побудови індивідуальних освітніх траєкторій вимогам технічного завдання.

5.2.2 Загальні положення

Випробування проводяться на основі наступних документів:

– ГОСТ 34.603–92. Інформаційна технологія. Види випробувань автоматизованих систем;

– ГОСТ РД 50-34.698-90. Автоматизовані системи вимог до змісту документів.

5.2.3 Результати випробувань

В процесі тестування були перевірена вся функціональність системи. У наступних таблицях приведено перелік випробувань основних функціональних можливостей (табл. 5.1 – 5.25).

Таблиця 5.1 – Уведення логіну та паролю

Мета тесту:	**Перевірка функції «Вхід в частину користувача»**
Початковий стан системи	Відкрита сторінка сайту «Home»
Вхідні дані:	Username або password користувача
Схема проведення тесту:	Натиснути кнопку «Log in».Ввести у поле «Username» ім’я користувача при реєстрації, а у поле «Password» ввести пароль користувача, натиснути кнопку «Log in»
Очікуваний результат:	Відкрита сторінка користувача
Стан КЗ після проведення випробувань:	Відкрита сторінка користувача

Таблиця 5.2 – Перевірка заповнення усіх атрибутів при авторизації

Мета тесту:	**Перевірка функції «Перевірка заповнення усіх атрибутів при авторизації»**
Початковий стан КЗ	Відкрита сторінка сайту «Home»
Вхідні данні:	Username або password
Схема проведення тесту:	Натиснути кнопку «Log in».Ввести у поле «Username» ім’я користувача при реєстрації, а у поле «Password»

Продовження таблиці 5.2

Мета тесту:	**Перевірка функції «Перевірка заповнення усіх атрибутів при авторизації»**
	ввести пароль користувача, натиснути кнопку «Log in»
Очікуваний результат:	Вікно повідомлення про помилку вводу. Вхід на сторінку користувача не виконано «Please enter a correct user name and password.»
Стан КЗ після проведення випробувань:	Відкрита сторінка авторизації

Таблиця 5.3 – Перевірка правильності введених даних при авторизації

Мета тесту:	**Перевірка функції «Перевірка правильності введених даних при авторизації»**
Початковий стан КЗ	Відкрита сторінка сайту «Home»
Вхідні данні:	Username або password
Схема проведення тесту:	Натиснути кнопку «Log in».Ввести у поле «Username» ім’я користувача при реєстрації, а у поле «Password» ввести пароль користувача, натиснути кнопку «Log in»
Очікуваний результат:	Вікно повідомлення про помилку вводу. Вхід на сторінку користувача не виконано «Please enter a correct user name and password.»
Стан КЗ після проведення випробувань:	Відкрита сторінка авторизації

Таблиця 5.4 – Повернення на головну сторінку сайту

Мета тесту:	**Перевірка функції «Повернення на головну сторінку сайту»**
Початковий стан КЗ:	Відкрита сторінка авторизації
Вхідні дані:	
Схема проведення тесту:	Натиснути кнопку «Home»
Очікуваний результат:	Відкрита головна сторінка сайту

Продовження таблиці 5.4

Мета тесту:	**Перевірка функції «Повернення на головну сторінку сайту»**
Стан КЗ після проведення випробування:	Відкрита головна сторінка сайту

Таблиця 5.5 – Вихід із користувацької сторінки

Мета тесту:	**Перевірка функції «Вихід із користувацької сторінки»**
Початковий стан КЗ:	Відкрита сторінка користувача
Вхідні дані:	
Схема проведення тесту:	Натиснути кнопку «Log off»
Очікуваний результат:	Відкрита головна сторінка сайту
Стан КЗ після проведення випробування:	Відкрита головна сторінка сайту

Таблиця 5.6 – Перехід на сторінку реєстрації

Мета тесту:	**Перевірка функції «Перехід на сторінку реєстрації»**
Початковий стан КЗ:	Відкрита сторінка авторизації
Вхідні дані:	
Схема проведення тесту:	Натиснути кнопку «Register here»
Очікуваний результат:	Відкрита сторінка реєстрації
Стан КЗ після проведення випробування:	Відкрита сторінка реєстрації

Таблиця 5.7 – Перевірка заповнення усіх атрибутів при реєстрації

Мета тесту:	**Перевірка функції «Перевірка заповнення усіх атрибутів при реєстрації»**
Початковий стан КЗ	Відкрита сторінка реєстрації
Вхідні данні:	User name, user surname, e-mail, username, password, role, для стейкхолдерів ще: your company name, company scope, your position, для тьюторів ще: input your field of activity or scope of work

Продовження таблиці 5.7

Мета тесту:	**Перевірка функції «Перевірка заповнення усіх атрибутів при реєстрації»**
Схема проведення тесту:	Натиснути кнопку «Register here».Ввести у поле «User name» ім’я користувача, у поле «User surname» прізвище користувача, у поле «E-mail» e-mail користувача, у поле «Username» унікальне ім’я користувача для входу у систему, у поле «Password» пароль користувача а виберіть роль. При реєстрації у ролі стекхолдера, додатково введіть у поле «Your company name» назву компанії, у якій працюєте, у поле «Company scope» профіль компанії, у поле «Your position» свою посаду. При реєстрації у ролі тьютора, додатково введіть у поле «Input your field of activity or scope of work» вид своєї діяльності. Натиснути кнопку «Register»
Очікуваний результат:	Повідомлення про помилку вводу.
Стан КЗ після проведення випробувань:	Відкрита сторінка реєстрації

Таблиця 5.8 – Перевірка правильності введених даних при реєстрації

Мета тесту:	**Перевірка функції «Перевірка правильності введених даних при реєстрації»**
Початковий стан КЗ	Відкрита сторінка реєстрації
Вхідні данні:	User name, user surname, e-mail, username, password, role, для стейкхолдерів ще: your company name, company scope, your position, для тьюторів ще: input your field of activity or scope of work
Схема проведення тесту:	Натиснути кнопку «Register here». Ввести у поле «User name» ім’я користувача, у поле «User surname» прізвище користувача, у поле «E-mail» e-mail користувача, у поле «Username» унікальне ім’я

Продовження таблиці 5.8

Мета тесту:	**Перевірка функції «Перевірка правильності введених даних при реєстрації»**
	користувача для входу у систему, у поле «Password» пароль користувача а виберіть роль. При реєстрації у ролі стекхолдера, додатково введіть у поле «Your company name» назву компанії, у якій працюєте, у поле «Company scope» профіль компанії, у поле «Your position» свою посаду. При реєстрації у ролі тьютора, додатково ввести у поле «Input your field of activity or scope of work» вид діяльності. Натиснути кнопку «Register»
Очікуваний результат:	Вікно повідомлення про помилку вводу
Стан КЗ після проведення випробувань:	Відкрита сторінка реєстрації

Таблиця 5.9 – Перехід на сторінку користувача

Мета тесту:	**Перевірка функції «Перехід на сторінку користувача»**
Початковий стан КЗ:	Відкрита будь-яка сторінка (користувач уже авторизований)
Вхідні дані:	
Схема проведення тесту:	Натиснути кнопку «My page»
Очікуваний результат:	Відкрита сторінка користувача
Стан КЗ після проведення випробування:	Відкрита сторінка користувача

Таблиця 5.10 – Перехід на сторінку створення траєкторії

Мета тесту:	**Перевірка функції «Перехід на сторінку створення траєкторії»**
Початковий стан КЗ:	Відкрита сторінка користувача ролі претендента
Вхідні дані:	

Продовження таблиці 5.10

Мета тесту:	**Перевірка функції «Перехід на сторінку створення траєкторії»**
Схема проведення тесту:	Натиснути кнопку «MAKE TRAJECTORY»
Очікуваний результат:	Відкрита сторінка формування траєкторії
Стан системи після проведення випробування:	Відкрита сторінка формування траєкторії

Таблиця 5.11 – Проходження психологічного тесту

Мета тесту:	**Перевірка функції «Проходження психологічного тесту»**
Початковий стан системи:	Відкрита сторінка формування траєкторії
Вхідні дані:	
Схема проведення тесту:	Натиснути кнопку «Psychological test»
Очікуваний результат:	Відкрита сторінка Google Forms для проходження психологічного тесту
Стан системи після проведення випробування:	Відкрита сторінка Google Forms для проходження психологічного тесту

Таблиця 5.12 – Проходження тесту на визначення лідерських якостей

Мета тесту:	**Перевірка функції «Проходження тесту на визначення лідерських якостей»**
Початковий стан КЗ:	Відкрита сторінка формування траєкторії
Вхідні дані:	
Схема проведення тесту:	Натиснути кнопку «Leadership test»
Очікуваний результат:	Відкрита сторінка Google Forms для проходження тесту на визначення лідерських якостей
Стан системи після проведення випробування:	Відкрита сторінка Google Forms для проходження тесту на визначення лідерських якостей

Таблиця 5.13 – Перехід на сторінку вибору професії

Мета тесту:	**Перевірка функції «Перехід на сторінку вибору професії»**
Початковий стан КЗ:	Відкрита сторінка формування траєкторії
Вхідні дані:	
Схема проведення тесту:	Натиснути кнопку «Proficiency&Competence test»
Очікуваний результат:	Відкрита сторінка вибору професії
Стан системи після проведення випробування:	Відкрита сторінка вибору професії

Таблиця 5.14 – Проходження тесту на професійні компетентності

Мета тесту:	**Перевірка функції «Проходження тесту на професійні компетентності»**
Початковий стан КЗ:	Відкрита сторінка сторінка вибору професії
Вхідні дані:	
Схема проведення тесту:	Натиснути кнопку «Go to the test»
Очікуваний результат:	Відкрита сторінка Google Forms для проходження тесту на професійні компетентності
Стан системи після проведення випробування:	Відкрита сторінка Google Forms для проходження тесту на професійні компетентності

Таблиця 5.15 – Перевірка мотивації

Мета тесту:	**Перевірка функції «Перевірка мотивації»**
Початковий стан КЗ:	Відкрита сторінка формування траєкторії
Вхідні дані:	
Схема проведення тесту:	Натиснути кнопку «Motivation detector»
Очікуваний результат:	Відкрита сторінка перевірки мотивації
Стан системи після проведення випробування:	Відкрита сторінка перевірки мотивації

Таблиця 5.16 – Формування траєкторії

Мета тесту:	**Перевірка функції «Формування траєкторії»**
Початковий стан КЗ:	Відкрита сторінка формування траєкторії

Продовження таблиці 5.16

Мета тесту:	**Перевірка функції «Формування траєкторії»**
Вхідні дані:	
Схема проведення тесту:	Натиснути кнопку «Make trajectory»
Очікуваний результат:	Дані претендента були надіслані тьютору для формування ним дисциплін, статус формування – «False». Відображення повідомлення про успішне надсилання даних: «Success! Your trajectory now is building!»
Стан системи після проведення випробування:	Відкрита сторінка формування траєкторії

Таблиця 5.17 – Перехід на сторінку визначення дисциплін

Мета тесту:	**Перевірка функції «Визначення дисциплін тьютором»**
Початковий стан КЗ:	Відкрита сторінка надісланих запитів на формування дисциплін користувача-тьютора
Вхідні дані:	Дані претендента
Схема проведення тесту:	Переглянути надіслані претендентами дані, натиснути на ім’я претендента
Очікуваний результат:	Відкрита сторінка визначення дисциплін
Стан системи після проведення випробування:	Відкрита сторінка визначення дисциплін

Таблиця 5.18 – Визначення дисцилін тьютором

Мета тесту:	**Перевірка функції «Визначення дисциплін тьютором»**
Початковий стан КЗ:	Відкрита сторінка визначення дисциплін
Вхідні дані:	Назви, кількість кредитів та форма контролю для восьми дисциплін

Продовження таблиці 5.18

Мета тесту:	**Перевірка функції «Визначення дисциплін тьютором»**
Схема проведення тесту:	Переглянути ключові слова до обраної претендентом професії. Ввести назви (відповідно до ключових слів), кількість кредитів та форма контролю для восьми дисциплін. Натиснути кнопку «Save disciplines»
Очікуваний результат:	Дисципліни було збережено до бази даних, статус формування дисциплін у претендента змінився на «True»
Стан системи після проведення випробування:	Відкрита сторінка надісланих запитів на формування дисциплін користувача-тьютора

Таблиця 5.19 – Перехід на сторінку вибору дисциплін претендентом

Мета тесту:	**Перевірка функції «Перехід на сторінку вибору дисциплін претендентом»**
Початковий стан КЗ:	Відкрита сторінка користувача ролі претендента
Вхідні дані:	
Схема проведення тесту:	Натиснути кнопку «CHOOSE DISCIPLINES»
Очікуваний результат:	Відкрита сторінка вибору дисциплін претендентом
Стан системи після проведення випробування:	Відкрита сторінка вибору дисциплін претендентом

Таблиця 5.20 – Вибір дисцилін претендентом

Мета тесту:	**Перевірка функції «Вибір дисциплін претендентом»**
Початковий стан КЗ:	Відкрита сторінка визначення дисциплін
Вхідні дані:	
Схема проведення тесту:	Обрати не менше 8 дисциплін. Натиснути кнопку «Choose!»

Продовження таблиці 5.20

Мета тесту:	**Перевірка функції «Вибір дисциплін претендентом»**
Очікуваний результат:	Обрані дисципліни було надіслано стейкхолдерові. статус затвердження – «Waiting for approving». Відображено повідомлення про успішний вибір: «Success! You have made your choice!»
Стан системи після проведення випробування:	Відкрита сторінка визначення дисциплін

Таблиця 5.21 – Перевірка коректності вибору кількості дисциплін претендентом

Мета тесту:	**Перевірка функції «Визначення дисциплін претендентом»**
Початковий стан КЗ:	Відкрита сторінка визначення дисциплін
Вхідні дані:	
Схема проведення тесту:	Обрати менше 8 дисциплін. Натиснути кнопку «Choose!»
Очікуваний результат:	Відображено повідомлення про не успішний вибір: «Please, choose 8!
Стан системи після проведення випробування:	Відкрита сторінка визначення дисциплін

Таблиця 5.22 – Перехід на сторінку затвердження дисциплін

Мета тесту:	**Перевірка функції «Перехід на сторінку затвердження дисциплін»**
Початковий стан КЗ:	Відкрита сторінка надісланих запитів на затвердження дисциплін користувача-стейкхолдера
Вхідні дані:	Дані претендента
Схема проведення тесту:	Переглянути надіслані претендентами дані, натиснути на ім’я претендента
Очікуваний результат:	Відкрита сторінка затвердження дисциплін
Стан системи після проведення випробування:	Відкрита сторінка затвердження дисциплін

Таблиця 5.23 – Затвердження дисциплін стейкхолдером

Мета тесту:	**Перевірка функції «Затвердження дисциплін стейкхолдером»**
Початковий стан КЗ:	Відкрита сторінка затвердження дисциплін
Вхідні дані:	Назви обраних восьми дисциплін та назва професії претендента
Схема проведення тесту:	Натиснути кнопку «Approve».
Очікуваний результат:	Дисципліни було затверджено та збережено до бази даних, статус затвердження дисциплін у претендента змінився на «Approved». Створилась нова індивідуальна. Відображення повідомлення про успішну операцію: «Your response was sent!»
Стан системи після проведення випробування:	Відкрита сторінка затвердження дисциплін

Таблиця 5.24 – Не затвердження дисциплін стейкхолдером

Мета тесту:	**Перевірка функції «Не затвердження дисциплін стейкхолдером»**
Початковий стан КЗ:	Відкрита сторінка затвердження дисциплін
Вхідні дані:	Назви обраних восьми дисциплін та назва професії претендента
Схема проведення тесту:	Натиснути кнопку «Not Approve».
Очікуваний результат:	Дисципліни не було затверджено та не було збережено до бази даних, статус затвердження дисциплін у претендента змінився на «Not Approved». Відображення повідомлення про успішну операцію: «Your response was sent!»
Стан системи після проведення випробування:	Відкрита сторінка затвердження дисциплін

Таблиця 5.25 – Перегляд сформованої індивідуальної траєкторії

Мета тесту:	**Перевірка функції «Перегляд сформованої індивідуальної траєкторії»**
Початковий стан КЗ:	Відкрита сторінка користувача ролі претендента
Вхідні дані:	
Схема проведення тесту:	Натиснути кнопку «MY TRAJECTORY».
Очікуваний результат:	Відкрита сторінка сформованої індивідуальної траєкторії
Стан системи після проведення випробування:	Відкрита сторінка сформованої індивідуальної траєкторії

Висновок до розділу

У даному розділі було описано випробування програмного продукту, а саме тести та порядок їх виконання для перевірки відповідності програмного забезпечення функціональним вимогам, які були представлені у технічному завданні на створення системи побудови індивідуальних освітніх траєкторій.

Для цього було визначено мету випробувань: перевірити відповідність функцій системи побудови індивідуальних освітніх траєкторій вимогам технічного завдання. Також визначено документи, на основі яких проводились випробування.

Результати випробувань представлено у табличному вигляді.

ЗАГАЛЬНІ ВИСНОВКИ

У ході виконання дипломного проекту були ґрунтовно розглянуті питання, які виникають в процесі створення індивідуальних освітніх траєкторій. Були виділені основні ключові етапи, притаманні процесу формування індивідуальних освітніх траєкторій, а також визначено взаємозв'язки між ними.

Був проведений докладний аналіз предметного середовища, ретельно описаний процес побудови індивідуальної траєкторії.

На основі даних, отриманих у процесі аналізу, було сформульовано відповідні математичні задачі. Створення індивідуальної траєкторії та подальше навчання не можливе без мотивації претендента. За рахунок аналізу мотиваційних листів та есе можна дізнатися мотивацію майбутнього студента. Також важливим етапом є визначення дисциплін для претендента. Це дозволить йому обрати відповідні дисципліни та створити індивідуальну траєкторію. Ці ключові етапи створення траєкторії було вирішено за допомогою математичних методів.

Для розробки програмного забезпечення була використана мова програмування Python, було обрано вільний фреймворк для веб-застосувань на мові Python, який має назву Django. Були використані такі засоби для програмування на Python як Visual Studio IDE з плагіном Python Tools for Visual Studio.

Була розроблена модель бази даних, яка надає можливість ефективно та надійно здійснювати доступ до даних, що використовуються в процесі формування індивідуальної траєкторії. Для управління базою даних обрана SQLite – база даних, яка легко вбудовується в додатки.

Наведена детальна інструкція користувача по експлуатації системи, описана методика проведення випробувань, яка показує можливість введення програми в експлуатацію.

ПЕРЕЛІК ПОСИЛАНЬ

1. ЗАКОН УКРАЇНИ Про освіту [Електронний ресурс] // Режим доступу: http://search.ligazakon.ua/l_doc2.nsf/link1/T172145.html

2. Розробка індивідуального освітнього маршруту для учнів 10-11 класів [Електронний ресурс] // Режим доступу: https://otherreferats.allbest.ru/pedagogics/00811694_0.html

3. Яснікова Н.С. Курсова робота «Індивідуальний освітній маршрут» [Електронний ресурс] // Режим доступу: https://infourok.ru/kursovaya-rabota-individualniy-obrazovatelniy-marshrut-497032.html

4. Чи актуально на сьогодні моделювання в IDEF0? [Електронний ресурс] // Режим доступу: http://projectimo.ru/biznes-processy/idef0.html#i-2

5. Інтерактивна мережа організацій та проектів [Електронний ресурс] // Режим доступу: http://www.xn--80akjhuieke7k.xn--80asehdb/

6. Система «Dean Dashboard» [Електронний ресурс] // Режим доступу: http://dean-dashboard.com/

7. Індивідуальні освітні траєкторії в Тюменському університеті [Електронний ресурс] // Режим доступу: https://www.utmn.ru/en/study-with-us/individual-education/

8. А.В. Проворова Індивідуальні освітні маршрути в основі особистої орієнтації учнів в умовах міжшкільного навчально комбінату [Електронний ресурс] // Режим доступу: https://lib.herzen.spb.ru/media/magazines/contents/1/102/provorova_102_254_258.pdf

9. Ф.Г. Мухаметзянова, Р.В. Забирова Проектування індивідуальної освітньої траєкторії і маршруту студента вузу-майбутнього бакалавра [Електронний ресурс] // Режим доступу: https://cyberleninka.ru/article/v/proektirovanie-individualnoy-obrazovatelnoy-traektorii-i-marshruta-studenta-vuza-buduschego-bakalavra

10. Н.М. Уварова, Т.В. Максимченко Індивідуальна освітня траєкторія як необхідна умова особистісно-професійного становлення студентів коледжу [Електронний ресурс] // Режим доступу: https://cyberleninka.ru/article/v/individualnaya-obrazovatelnaya-traektoriya-kak-neobhodimoe-uslovie-lichnostno-professionalnogo-stanovleniya-studentov-kolledzha

11. О.В. Гончарова, Р.М. Чумичева Проблеми і перспективи сучасної освіти [Електронний ресурс] // Режим доступу: https://cyberleninka.ru/article/v/organizatsiya-individualnoy-obrazovatelnoy-traektorii-obucheniya-bakalavrov

12. Застосування методу латентно-семантичний аналіз для автоматичної рубрикації документів [Електронний ресурс] // Режим доступу: https://cyberleninka.ru/article/n/primenenie-metoda-latentno-semanticheskogo-analiza-dlya-avtomaticheskoy-rubrikatsii-dokumentov

13. Латентно-семантичний аналіз тексту [Електронний ресурс] // Режим доступу: http://izron.ru/articles/aktualnye-problemy-tekhnicheskikh-nauk-v-rossii-i-za-rubezhom-sbornik-nauchnykh-trudov-po-itogam-mezh/sektsiya-2-informatika-vychislitelnaya-tekhnika-i-upravlenie-spetsialnost-05-13-00/latentno-semanticheskiy-analiz-teksta/

14. Башарін Є.В. Контекстна обробка даних соціальних мереж [Електронний ресурс] // Режим доступу: https://nauchkor.ru/uploads/documents/587d36355f1be77c40d5896c.pdf

15. Гурко М.В. Розробка веб-сервісу рекомендації книг на основі тематичного моделювання [Електронний ресурс] // Режим доступу: http://elib.spbstu.ru/dl/2/v17-5253.pdf/download/v17-5253.pdf

16. Машинне навчання. Тематичне моделювання [Електронний ресурс] // Режим доступу: http://edu.mmcs.sfedu.ru/pluginfile.php/18218/mod_resource/content/2/11%20%D0%A2%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D1%87%D0%B5%D1

%81%D0%BA%D0%BE%D0%B5%20%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5.pdf

17. Візуалізація результатів латентно-семантичного аналізу засобами Python [Електронний ресурс] // Режим доступу: https://habr.com/post/335668/

18. Чому Python? [Електронний ресурс] // Режим доступу: https://khashtamov.com/ru/why-python/

19. Python: переваги и недоліки [Електронний ресурс] // Режим доступу: https://www.goldenpages.ua/expert/?p=10627

20. Застосування і переваги мови програмування Python [Електронний ресурс] // Режим доступу: http://securos.org.ua/primenenie-i-preimushhestva-yazyika-programmirovaniya-python/

21. SQLite vs MySQL vs PostgreSQL: порівняння систем керування базами даних [Електронний ресурс] // Режим доступу: http://devacademy.ru/posts/sqlite-vs-mysql-vs-postgresql/

22. Django Book: концепція розробки MVC [Електронний ресурс] // Режим доступу: https://rtfm.co.ua/django-book-model-razrabotki-mtc-model-view-controller/

23. Вступ до Django [Електронний ресурс] // Режим доступу: https://developer.mozilla.org/ru/docs/Learn/Server-side/Django/%D0%92%D0%B2%D0%B5%D0%B4%D0%B5%D0%BD%D0%B8%D0%B5

24. Психологічний тест [Електронний ресурс] // Режим доступу: https://forms.gle/Juf7nWPRPjd8aWLe9

25. Тест на визначення лідерських якостей [Електронний ресурс] // Режим доступу: https://forms.gle/7BWiDbbetF73drW68

26. Тест на професійні компетентності [Електронний ресурс] // Режим доступу: https://cepisecompetencebenchmark.org/

Додаток А

Тексти програмного коду

Інформаційна система побудови індивідуальних освітніх траєкторій

(Найменування програми (документа))

DVD-R

(Вид носія даних)

12 арк, 402,7 МБ

(Обсяг програми (документа) , арк.,) Кб)

Київ – 2019 року

Реалізація методу латентно-семантичного аналізу тексту для визначення мотивації претендентів (MyAlgo.py)

```
"""description of class"""
from mpi4py import MPI
import Dvornyk_IS52
import time
import nltk
import imp
import numpy
import numpy as np
from math import sqrt
from nltk.corpus import stopwords
from nltk.stem import SnowballStemmer
import numpy as np
import threading
import random
stemmer = SnowballStemmer("english")
class LSI(object):
    def __init__(self, stopwords, ignorechars, docs):

        self.wdict = {}
              # dictionary - key words
        self.dictionary = []
        self.fulldictionary = []
              # stopwords
        self.stopwords = stopwords
        self.docs = []
        if type(ignorechars) == str: ignorechars = ignorechars.encode('utf-8')
        self.ignorechars = ignorechars
              # init docs
        for doc in docs: self.add_doc(doc)
    def prepare(self):
        self.build()
        self.calc()
    def dic(self, word, add = False):
        if type(word) == str: word = word.encode('utf-8')
        word = word.lower().translate(None, self.ignorechars)
        word = word.decode('utf-8')
        fullword = word
        word = stemmer.stem(word)
              # checking stopwords
        if word not in self.stopwords:
              if word in self.dictionary: return self.dictionary.index(word)
              else:
                     if add:
                            self.dictionary.append(word)
                            self.fulldictionary.append(fullword)
                            return len(self.dictionary) - 1
                     else: return None
        else: return -1

    def add_doc(self, doc):
        words = [self.dic(word, True) for word in doc.lower().split()]
        words = [x for x in words if x != -1]
        self.docs.append(words)
        for word in words:
              if self.dictionary[word] in self.stopwords:
                     continue
```

```
            elif word in self.wdict:   self.wdict[word].append(len(self.docs) - 1)
            else:                      self.wdict[word] = [len(self.docs) - 1]
    def build(self):
        # remove words that was used once
        self.keys = [k for k in self.wdict.keys() if len(self.wdict[k]) > 0]
        self.keys.sort()
        self.A = numpy.zeros([len(self.keys), len(self.docs)])
        for i, k in enumerate(self.keys):
            for d in self.wdict[k]:
                # how many times word was in doc
                self.A[i,d] += 1

    def calc(self):
            # singular (U, S Vt)
        self.U, self.S, self.Vt = numpy.linalg.svd(self.A)

       #schema tfidf
    def TFIDF(self):
        wordsPerDoc = numpy.sum(self.A, axis=0)
        docsPerWord = numpy.sum(numpy.asarray(self.A > 0, 'i'), axis=1)
        rows, cols = self.A.shape
        for i in range(rows):
            for j in range(cols):
                self.A[i,j] = numpy.round((self.A[i,j] / wordsPerDoc[j]) *
numpy.log(float(cols) / docsPerWord[i]), 3)

    def dump_src(self):
        self.prepare()
        self.TFIDF()
        print('Матрица TF-IDF частот слов в документах без стоп слов (указаны полные
версии слов без стемминга)')
        for i, row in enumerate(self.A):
            print (row, self.fulldictionary[self.keys[i]])

    def print_svd(self):
        self.prepare()
        self.TFIDF()

    def find(self, word):
        count = 0
        self.prepare()
        idx = -1
        idx = self.dic(word)
        if idx == -1:
            return []
        if not idx in self.keys:
            return []
        idx = self.keys.index(idx)
        wx, wy = (self.U[:, 1:3])[idx]
        arts = []
        xx, yy = self.Vt[1:3, :]
        for k, v in enumerate(self.docs):
            if idx in v: inc = "СОВПАДЕНИЕ"
            else: inc = ""
            ax, ay = xx[k], yy[k]
            dx, dy = float(wx - ax), float(wy - ay)
            arts.append((k, v, ax, ay, 1-
float(wx*ax+wy*ay)/float(sqrt(wx*wx+wy*wy)*sqrt(ax*ax+ay*ay)), inc))
        return sorted(arts, key = lambda a: a[4])
class MyAlgo(object):
    docs = []
    words = []
```

```
    def __init__(self, docs, words):
        self.docs = docs
        self.words = words

    if len(docs) == 0:
    def do_work(self):
        punctuation = ''',:;'!'''
        from nltk.corpus import stopwords
        stopwords = stopwords.words('english')
        if (len(self.words)==0):
            self.words =
["accounting","administrator","advance","analysis","analysts","analyze","annual",

       "application","architect","associate","background","business","carpal","carrier","
certification","chapter",

       "chief","code","common","communicate","communication","competitive","computer","co
mputing","concentrate","considerable","consultant",

       "consulting","coordinate","create","customer","cyber","data","database","deal","de
cline","design","designer","detailed",

       "developer","development","performance","problem","product","program","programmer"
,"project","reduce","resource",

       "science","security","software","site","software","task","system","technical","tec
hnician","technological","telecommunications",

       "transfer","web","webmaster","worker","workplace","effectively","efficiency","elec
tronic","employ","engineering","engineer","enterprise","environment","equipment","experti
se","eyestrain",

       "finance","function","goal","graduate","hardware","install","instruction","integra
te","intranet",

       "introductory","involved","keyboard","knowledge","language","level","location","ma
intain","maintenance",

       "marketing","mathematics","matrix","median","mobile","monitor","network","networki
ng","office","offshore","order","outsourcing",
                    "pdf","perform"
           ]
        array_res = numpy.zeros(1)
        sum = numpy.zeros(1)
              #paralleling
        comm = MPI.COMM_WORLD
        size = comm.Get_size()
        rank = comm.Get_rank()
        word = ""
        if rank == 0:
           chunks = [[] for _ in range(size)]
           for i, chunk in enumerate(self.words):
                chunks[i % size].append(chunk)
           for i in range(size):
               lsa = LSI(stopwords, punctuation, self.docs)
               k=i+2
               data=self.words[i]
           #print ('we will be scattering:',word)
        else:
           data = None
           chunks = None
```

```
        task_arg = 0
        data = comm.scatter(chunks, root=0)
        task = LSI([], punctuation, self.docs)
        task.build()
        task.dump_src()
        task.calc()
        task.print_svd()
        task.words = data
        res1 = []
        areas = []
        v = 0
        b = 0
        c = 0
        for word in data:
            myres = task.find(word)
            if not myres:
                c=len(self.docs)*2
            for res in task.find(word):
                if res[4]<0.5:
                    v+=abs(res[4])
                if res[4]<0.0001:
                    if self.docs[res[0]] not in areas:
                        areas.append(self.docs[res[0]])
                print (res[0], res[4], res[1], self.docs[res[0]])
                res1.append(res[0])
                res1.append(res[4])
                res1.append(res[1])
                res1.append(self.docs[res[0]])
                #print(v)
            print ("For words", v)
            b+=(v+c)
            v=0
            c=0
            print ("B", b)
            array_res[0] = b
        comm.Reduce(array_res, sum, MPI.SUM, 0)
        answer = []
        answer.append(str(sum[0]))
        answer.append(areas)
        if rank == 0:
            print(sum[0])
        return answer
```

Реалізація методу латентного розміщення Діріхле для визначення ключових слів областей знань (views.py)

```
def getAreas(user):
    competencies = ""
    applicant = Applicant.objects.get(userid=user.id)
    job = Job.objects.get(id=applicant.jobid)
    compInJob = CompJob.objects.all().filter(jobid=job.id)
    for item in compInJob:
        competencies += Competence.objects.get(id=item.competenceid).name
        competencies += ' '
        competencies += Competence.objects.get(id=item.competenceid).knowledge
        competencies += ' '
        competencies +=Competence.objects.get(id=item.competenceid).skill
    list_text = []
    algo = Algorithm()
    # competencies as words
    tokenizer = RegexpTokenizer('[A-Za-z]\w+')
```

```
    result = tokenizer.tokenize(competencies)
    new_sentence =[]
    for w in result:
        if w not in stopwords: new_sentence.append(w)
    # areas ad docs
    areasDocs = []
    areas = KnowledgeArea.objects.all()
    for area in areas:
        areasDocs.append(area.name + ' ' + area.topics)
    algo.text = areasDocs
    algor = MyAlgo(algo.text, new_sentence)
    resultAreas = algor.do_work()
    # find keyword
    areaskey = ""
    for area in resultAreas[1]:
        areaskey += area

    try:
        jobinareas = JobsInAreas.objects.all()
    except JobsInAreas.DoesNotExist:
        for area in resultAreas[1]:
            areasfromdb = KnowledgeArea.objects.all()
            for areafromdb in areasfromdb:
                if areafromdb.name in area:
                    jobinareas = JobsInAreas()
                    jobinareas.jobid = job.id
                    jobinareas.areaid = areafromdb.id
                    jobinareas.save()
    return areaskey
```

Реалізація реєстрації (views.py)

```
def register(request):
    user = User()
    stakeholder = Stakeholder()
    tutor = Tutor()
    try:
        error = ""
        if request.method == "POST":
            user = User()
            user.name = request.POST.get("name")
            user.surname = request.POST.get("surname")
            user.useremail = request.POST.get("useremail")
            user.username = request.POST.get("username")
            user.password = request.POST.get("password")
            user.role = request.POST.get("role")
            user.save()

            user1 = userdj()
            user1.email = request.POST.get("useremail")
            user1.username = request.POST.get("username")
            user1.set_password(request.POST.get("password"))
            user1.save()

            if user.role == "stakeholder":
                stakeholder = Stakeholder()
                stakeholder.companyName = request.POST.get("companyName")
                stakeholder.scopeCompany = request.POST.get("scopeCompany")
                stakeholder.position = request.POST.get("position")
                stakeholder.userid = user.id
                stakeholder.save()
```

```
            if user.role == "tutor":
                tutor = Tutor()
                tutor.scope = request.POST.get("scope")
                tutor.userid = user.id
                tutor.save()
        return render(request,
        'app/register.html',
        {
            'title':'Register',
            'message':'Your register page.',
            'year':datetime.now().year,
            "error":"",
            "user":user,
            "stakeholder":stakeholder,
            "tutor":tutor
        })
    except IntegrityError as e:
        return render(request,
        'app/register.html',
        {
            'title':'Register',
            'message':'Your register page.',
            'year':datetime.now().year,
            "error":"This username already exists",
            "user":user,
            "stakeholder":stakeholder,
            "tutor":tutor
        })
```

Реалізація визначення результатів тестів (views.py)

```
def parseSheets(applicant,user):
    iduser = 0
    iduser2 = 0

    for i in range(sheet.row_count - 100):
        i = i + 1
        email = sheet.cell(i,2).value
        if (email == user.useremail):
            iduser = i
    for i in range(sheet2.row_count - 100):
        i = i + 1
        email = sheet2.cell(i,2).value
        if (email == user.useremail):
            iduser2 = i
    if(iduser == 0 or iduser2 == 0):
        applicant.creation = 0
        applicant.social = 0
        applicant.tech = 0
        applicant.nature = 0
        applicant.test1 = 0
        applicant.datetest1 = 0
        applicant.test2 = 0
        applicant.datetest2 = 0
        applicant.test3 = 0
        applicant.motivation = 0
        applicant.jobid = 0
        applicant.appliedForTraj = 0
        applicant.approvedTraj = 0
        applicant.save()
```

```
    return applicant
# test 1
creation = 0
social = 0
tech = 0
nature = 0
if (sheet.cell(iduser,3).value == answers[0]):
    creation = creation + 1
else:
    social = social + 1
if (sheet.cell(iduser,4).value == answers[1]):
    tech = tech + 1
else:
    creation = creation + 1
if (sheet.cell(iduser,5).value == answers[2]):
    tech = tech + 1
else:
    creation = creation + 1
if (sheet.cell(iduser,6).value == answers[3]):
    tech = tech + 1
else:
    social = social + 1
if (sheet.cell(iduser,7).value == answers[4]):
    tech = tech + 1
else:
    nature = nature + 1
if (sheet.cell(iduser,8).value == answers[5]):
    tech = tech + 1
else:
    nature = nature + 1
if (sheet.cell(iduser,9).value == answers[6]):
    creation = creation + 1
else:
    nature = nature + 1
if (sheet.cell(iduser,10).value == answers[7]):
    nature = nature + 1
else:
    social = social + 1
if (sheet.cell(iduser,11).value == answers[8]):
    tech = tech + 1
else:
    nature = nature + 1
if (sheet.cell(iduser,12).value == answers[9]):
    creation = creation + 1
else:
    social = social + 1
if (sheet.cell(iduser,13).value == answers[10]):
    nature = nature + 1
else:
    social = social + 1
if (sheet.cell(iduser,14).value == answers[11]):
    social = social + 1
else:
    creation = creation + 1
applicant.creation = round(creation * 100 / 6)
applicant.social = round(social * 100 / 6)
applicant.tech = round(tech * 100 / 6)
applicant.nature = round(nature * 100 / 6)
max = creation
maxval = "creation"
if(max < social):
    max = social
```

```
            maxval = "social"
        elif(max < tech):
            max = tech
            maxval = "tech"
        elif(max < nature):
            max = nature
            maxval = "nature"
        applicant.test1 = maxval
        applicant.datetest1 = sheet.cell(iduser,1).value
        # test 2
        test2answers = ["Not at All", "Sometimes", "Very Often"]
        test2result = 0
        test2res = ""
        k = 3
        while k <= 12:
            if (sheet2.cell(iduser2,k).value == test2answers[0]):
                test2result += 1
            elif (sheet2.cell(iduser2,k).value == test2answers[1]):
                test2result += 3
            else:
                test2result += 5
            k+=1
        applicant.test2 = round(test2result * 100 / 50)
        applicant.datetest2 = sheet2.cell(iduser2,1).value
        applicant.test3 = 0
        applicant.motivation = 0
        applicant.jobid = 0
        applicant.appliedForTraj = 0
        applicant.approvedTraj = 0
        applicant.save()
        if (test2result >= 10 and test2result < 20):
            test2res = "You need to work hard on your leadership skills, you're " +
    str(test2result * 100 / 50) + "% leader!"
        elif (test2result >= 20 and test2result < 40):
            test2res = "You're " + str(test2result * 100 / 50) + "% leader, it's OK, but you
    have the potential to do much better!"
        else:  test2res = "Excellent! You're " + str(test2result * 100 / 50) + "% leader and
    you're on your way to becoming a good leader!"
        print(list_of_hashes)
        print(list_of_hashes2)
        return applicant
    def trajectoriesApp(request, username):
        try:
            user = User.objects.get(username=username)
            user1 = userdj.objects.get(username=username)
            if request.method == "POST":
                applicant = Applicant.objects.get(userid=user.id)
                applicant.appliedForTraj = True
                applicant.save()
                keywords = getAreas(user)
                messages.success(request, 'Your trajectory now is building!')
                #return render(request, "app/applicant.html", {"user": user1, "user1": user})
            try:
                applicant = Applicant.objects.get(userid=user.id)
                idcell1 = 0
                idcell2 = 0
                for i in range(sheet.row_count - 100):
                    i = i + 2
                    email1 = sheet.cell(i,2).value
                    if (email1 == user.useremail):
                        idcell1 = i
                for i in range(sheet2.row_count - 100):
```

```
                i = i + 2
                email2 = sheet2.cell(i,2).value
                if (email2 == user.useremail):
                    idcell2 = i
            if(idcell1 == 0 or idcell2 == 0):
                applicant.creation = 0
                applicant.social = 0
                applicant.tech = 0
                applicant.nature = 0
                applicant.test1 = 0
                applicant.datetest1 = 0
                applicant.test2 = 0
                applicant.datetest2 = 0
                applicant.test3 = 0
                applicant.motivation = 0
                applicant.jobid = 0
                applicant.appliedForTraj = 0
                applicant.approvedTraj = 0
                applicant.save()
                try:
                    job = Job.objects.get(id=applicant.jobid)
                except Job.DoesNotExist:
                    job = Job()
                    job.name = "No job suggested"
                return render(request, "app/trajectoriesApp.html", {"user": user1,
"user1": user, "nature":applicant.nature, "tech":applicant.tech,

"social":applicant.social, "creation":applicant.creation,
                                                                    "test2res": "Please,
take a test to have result", "job":job,

"motivation":applicant.motivation, "applicant":applicant
                                                                })
            if(applicant.datetest1 != sheet.cell(idcell1,1).value or applicant.datetest2
!= sheet2.cell(idcell2,1).value):
                applicant = parseSheets(applicant,user)
            if(applicant.test1 != None or applicant.test2 != None):
                if (applicant.test2 >= 20 and applicant.test2 < 40):
                    test2res = "You need to work hard on your leadership skills, you're "
+ str(applicant.test2) + "% leader!"
                elif (applicant.test2 >= 40 and applicant.test2 < 80):
                    test2res = "You're " + str(applicant.test2) + "% leader, it's OK, but
you have the potential to do much better!"
                else:  test2res = "Excellent! You're " + str(applicant.test2) + "% leader
and you're on your way to becoming a good leader!"
                try:
                    job = Job.objects.get(id=applicant.jobid)
                except Job.DoesNotExist:
                    job = Job()
                    job.name = "No job suggested"
                return render(request, "app/trajectoriesApp.html", {"user": user1,
"user1": user, "nature":applicant.nature, "tech":applicant.tech,

"social":applicant.social, "creation":applicant.creation,
                                                                    "test2res": test2res,
"job":job,

"motivation":applicant.motivation, "applicant":applicant
                                                                })
        except Applicant.DoesNotExist:
            applicant = Applicant()
            applicant.userid = user.id
```

```
            try:
                applicant = parseSheets(applicant,user)
            except User.DoesNotExist:
                return render(request, "app/trajectoriesApp.html", {"user": user1,
"user1": user, "nature":applicant.nature, "tech":applicant.tech,

"social":applicant.social, "creation":applicant.creation,
                                                                    "test2res": test2res,
"job": job,

"motivation":applicant.motivation, "applicant":applicant})
            if (applicant.test2 >= 20 and applicant.test2 < 40):
                test2res = "You need to work hard on your leadership skills, you're " +
str(applicant.test2) + "% leader!"
            elif (applicant.test2 >= 40 and applicant.test2 < 80):
                test2res = "You're " + str(applicant.test2) + "% leader, it's OK, but you
have the potential to do much better!"
            else:  test2res = "Excellent! You're " + str(applicant.test2) + "% leader and
you're on your way to becoming a good leader!"
            try:
                job = Job.objects.get(id=applicant.jobid)
            except Job.DoesNotExist:
                job = Job()
                job.name = "No job suggested"
            return render(request, "app/trajectoriesApp.html", {"user": user1, "user1":
user, "nature":applicant.nature, "tech":applicant.tech,

"social":applicant.social, "creation":applicant.creation,
                                                                    "test2res": test2res,
"job": job,

"motivation":applicant.motivation, "applicant":applicant})
    except User.DoesNotExist:
        return HttpResponseNotFound("<h2>User not found</h2>")
    assert isinstance(request, HttpRequest)
    return render(request,
        'app/trajectoriesApp.html',
        {
            'title':'Contact',
            'message':'Your contact page.',
            'year':datetime.now().year,
        })
```

Реалізація визначення мотивації (views.py)

```
def analyzer(request, username):
    try:
        user = User.objects.get(username=username)
        user1 = userdj.objects.get(username=username)
        applicant = Applicant.objects.get(userid=user.id)
        job = Job.objects.get(id=applicant.jobid)
        list_text = []
        analyzerform = AnalyzerForm()
        """Renders the about page."""
        if request.method == "POST":
            algo = Algorithm()
            algo.text = request.POST.get("text")
            wordCount = len(str(algo.text).split())
            tokenizer = nltk.data.load('tokenizers/punkt/english.pickle')
            print('\n-----\n'.join(tokenizer.tokenize(str(algo.text))))
```

```
            list_text.append(tokenizer.tokenize(str(algo.text)))
            algo.text = list_text[0]
            algor = MyAlgo(algo.text, "")
            start_time = time.time()
            res = algor.do_work()
            end_time = time.time()
            print(list_text[0])
            tokenized_texts = [list(tokenize(text1.lower())) for text1 in list_text[0]]
            words_reviews = []

            for review in tokenized_texts:
                wr = []
                for word in review:
                    if word not in stopwords:
                        wr.append(word)
                words_reviews.append(wr)
            dictionary = Dictionary(words_reviews)
            corpus = [dictionary.doc2bow(text1) for text1 in list_text]
            model = models.LdaMulticore(corpus=corpus, id2word=dictionary, num_topics=1,
workers = 3)
            topics = model.show_topics()
            themes = str(topics)
            result = []
            for topic in topics:
                result.append(re.sub('[^a-z]', ' ', str(topic)))
                themes += "".join(str(topic))
            temp = 2 * (len(algo.text) / 2) * 108 * 0.95
            resi = float(res[0]) / 2
            if resi < temp:
                answer = True
            else: answer = False
            #result = re.sub('[^a-z]', '', themes)
            applicant.motivation = answer
            applicant.save()
            if (applicant.test2 >= 20 and applicant.test2 < 40):
                    test2res = "You need to work hard on your leadership skills, you're "
+ str(applicant.test2) + "% leader!"
            elif (applicant.test2 >= 40 and applicant.test2 < 80):
                    test2res = "You're " + str(applicant.test2) + "% leader, it's OK, but
you have the potential to do much better!"
            else:  test2res = "Excellent! You're " + str(applicant.test2) + "% leader and
you're on your way to becoming a good leader!"
            return render(request, "app/trajectoriesApp.html", {"user": user1, "user1":
user, "nature":applicant.nature, "tech":applicant.tech,

"social":applicant.social, "creation":applicant.creation,
                                                                    "test2res": test2res,
"job": job,

"motivation":applicant.motivation, "applicant":applicant})
        else:
            assert isinstance(request, HttpRequest)
            return render(request,
                'app/analyzer.html',
                {
                    'title':'Analyzer',
                    'message':'Your analyzer page.',
                    'year':datetime.now().year,
                    'form' : analyzerform
                })
    except User.DoesNotExist:
        return HttpResponseNotFound("<h2>User not found</h2>")
```

Реалізація вибору дисциплін претендентом (views.py)

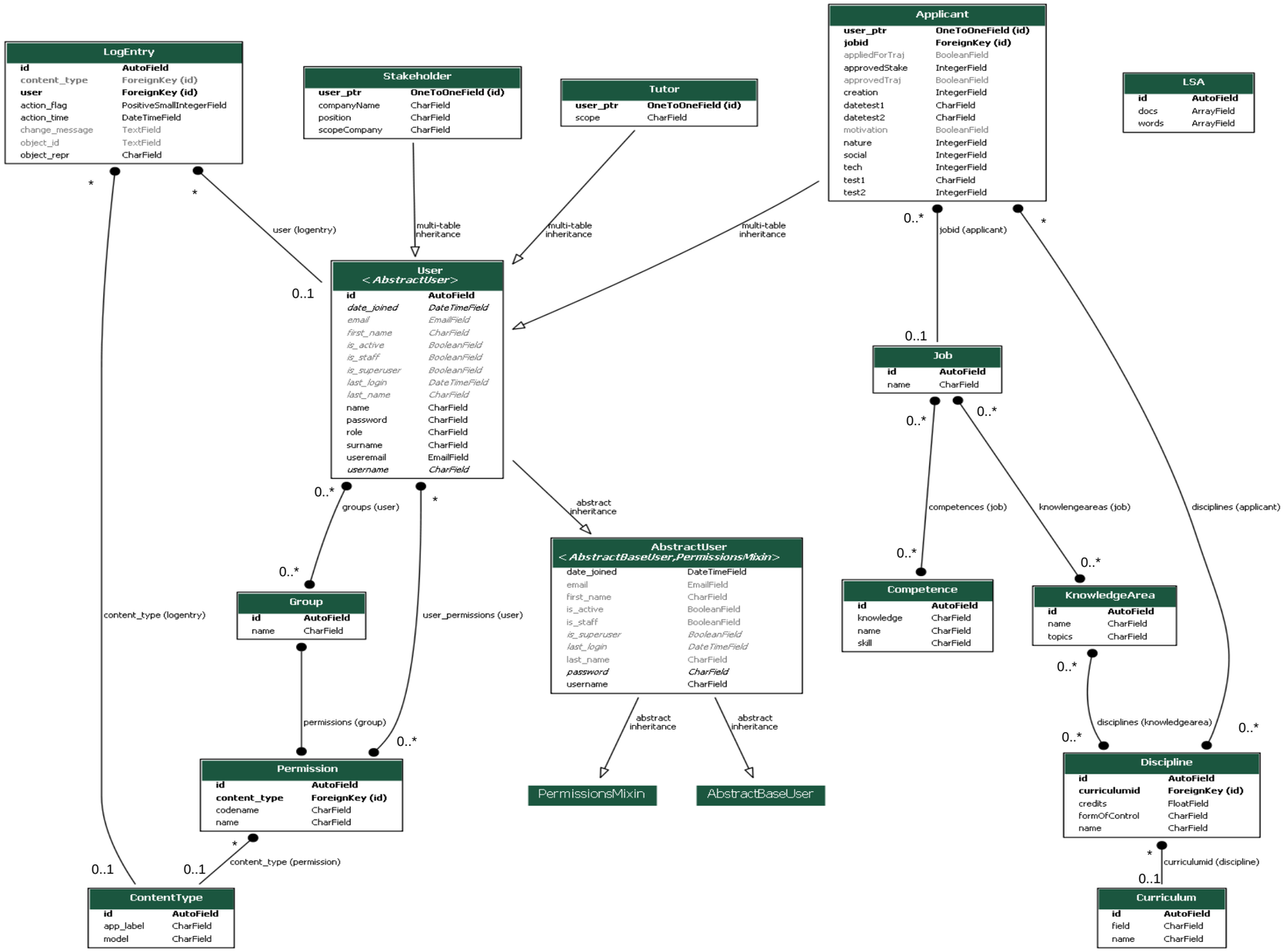
```
def listTrajApp(request, username):
    try:
        disc = []
        d = []
        areas1 = []
        disciplinesinarea1 = []
        user = User.objects.get(username=username)
        user1 = userdj.objects.get(username=username)
        applicant = Applicant.objects.get(userid=user.id)
        job = Job.objects.get(id=applicant.jobid)
        jobsinareas = JobsInAreas.objects.all().filter(jobid=applicant.jobid)
        areas = KnowledgeArea.objects.all()
        for jobsinarea in jobsinareas:
            for area in areas:
                if(area.id == jobsinarea.areaid):
                    areas1.append(area)
        disciplinesinarea = DisciplinesInAreas.objects.all()
        for area in areas1:
            for disciplineinarea in disciplinesinarea:
                if area.id == disciplineinarea.areaid:
                    disciplinesinarea1.append(disciplineinarea)
        disciplines = Discipline.objects.all()
        for disciplineinarea in disciplinesinarea1:
            for discipline in disciplines:
                if disciplineinarea.disciplineid == discipline.id:
                        disc.append(discipline)
        lenght = int(len(disc)/len(areas1))
        d1 = disc [:lenght]
        if request.method == "POST":
            disciplines = Discipline.objects.all()
            disc = []
            for d in request.POST.getlist('discipline'):
                for discipline in disciplines:
                    if discipline.id == int(d):
                        disc.append(discipline)
            if (len(disc) < 8):
                messages.error(request, 'Please, choose 8!')
            else:
                if
(DisciplineForApplicant.objects.all().filter(applicantid=applicant.id)):

DisciplineForApplicant.objects.all().filter(applicantid=applicant.id).delete()
                for discipline in disc:
                    disciplineforapplicant = DisciplineForApplicant()
                    disciplineforapplicant.disciplineid = discipline.id
                    disciplineforapplicant.applicantid = applicant.id
                    disciplineforapplicant.save()
                    applicant.approvedStake = 0
                    applicant.save()

                messages.success(request, 'You have made your choice!')
        return render(request, "app/listTrajApp.html", {"user": user1, "user1": user,
"disciplines": d1, "applicant": applicant})
    except User.DoesNotExist:
         return render(request, "app/listTrajApp.html", {"user": user1, "user1": user})
```

Реалізація затвердження стейкхолдером дисциплін (views.py)

```
def approveStakeDisciplines(request, username, applicantusername):
    try:
        disc = []
        user = User.objects.get(username=username)
        user1 = userdj.objects.get(username=username)
        applicants = Applicant.objects.all().filter(appliedForTraj=True,
approvedTraj=True, approvedStake = 0)
        for applicant in applicants:
            userapplicants = User.objects.all().filter(id=applicant.userid)
        applicantuser = User.objects.get(username=applicantusername)
        app = Applicant.objects.get(userid=applicantuser.id)
        job = Job.objects.get(id = app.jobid)
        disiplinesforapplicant = DisciplineForApplicant.objects.all().filter(applicantid
= app.id)
        disciplines = Discipline.objects.all()
        for disiplineforapplicant in disiplinesforapplicant:
            for discipline in disciplines:
                if(discipline.id == disiplineforapplicant.disciplineid):
                    disc.append(discipline)
        if request.method == "POST":
            value = request.POST.get("btnAddDisciplines")
            if value == "Approve":
                app.approvedStake = 1
                app.save()
            elif value == "Not Approve":
                app.approvedStake = -1
                app.save()
            messages.success(request, 'Your response was sent!')
        return render(request, "app/approveStakeDisciplines.html", {"user": user1,
"user1": user, "disciplines": disc, "job": job})
    except User.DoesNotExist:
         return render(request, "app/approveStakeDisciplines.html", {"user": user1,
"user1": user})
```

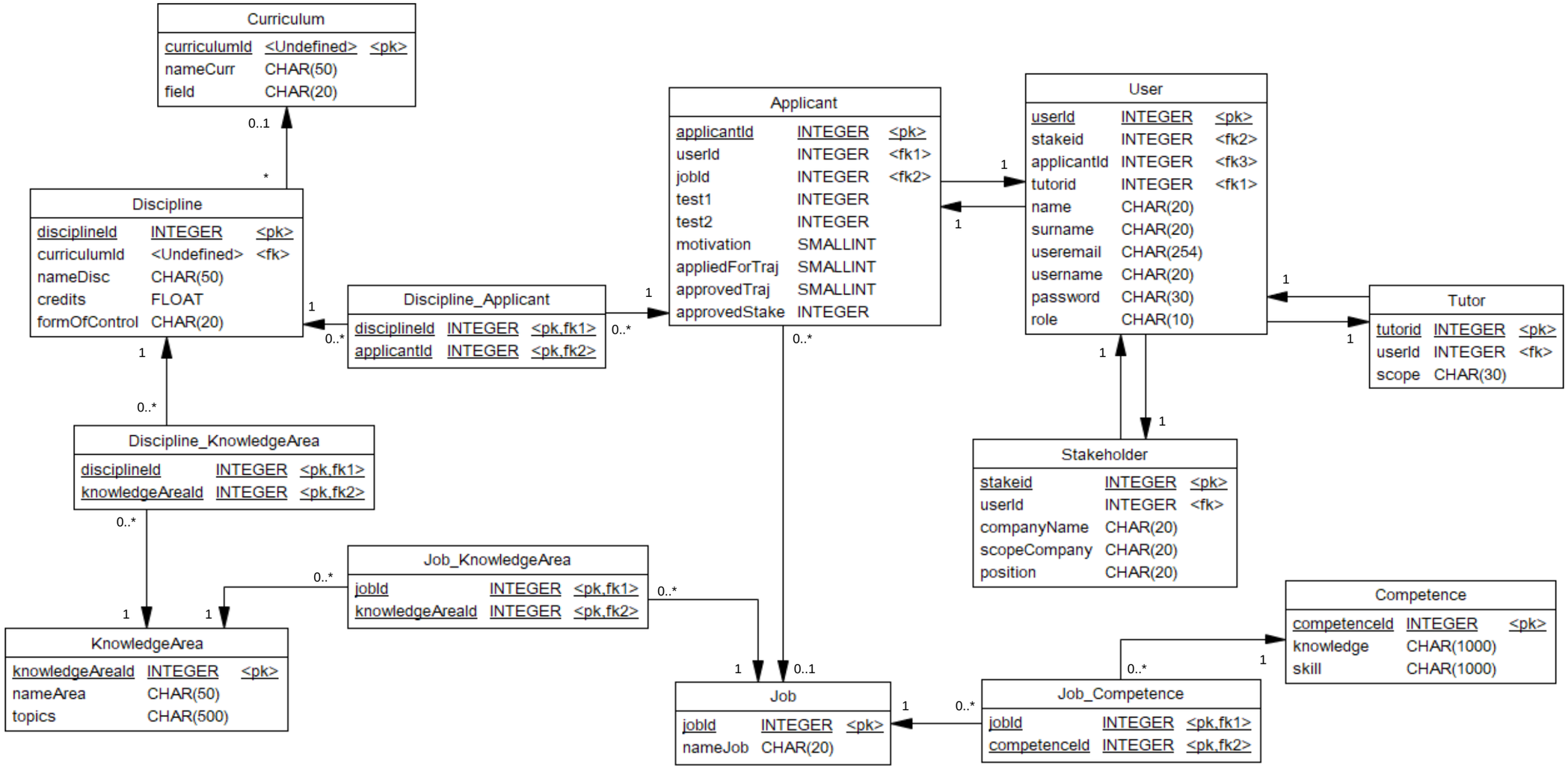
					ДП ІС-5207.1181-с.ССД	
Зм.	Арк.	№ документа	Підпис	Дата	Схема структурна діяльності	Літера / Маса / Масштаб
Розробив		Дворник В.А.				
Перевірив		Ковалюк Т.В.				Аркуш 1 / Аркушів 1
Т. кон.					Інформаційна система побудови індивідуальних освітніх траєкторій	КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-52
Н. кон.		Халус О.А.				
Затвердив		Ковалюк Т.В.				

ДП ІС-5207.1181-с.ССК



					ДП ІС-5207.1181-с.ССК	
Зм.	Арк.	№ документа	Підпис	Дата	Схема структурна класів	Літера / Маса / Масштаб
Розробив		Дворник В.А.				
Перевірив		Ковалюк Т.В.				Аркуш 1 / Аркушів 1
Т. кон.						
Н. кон.		Халус О.А.			Інформаційна система побудови індивідуальних освітніх траєкторій	КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-52
Затвердив		Ковалюк Т.В.				

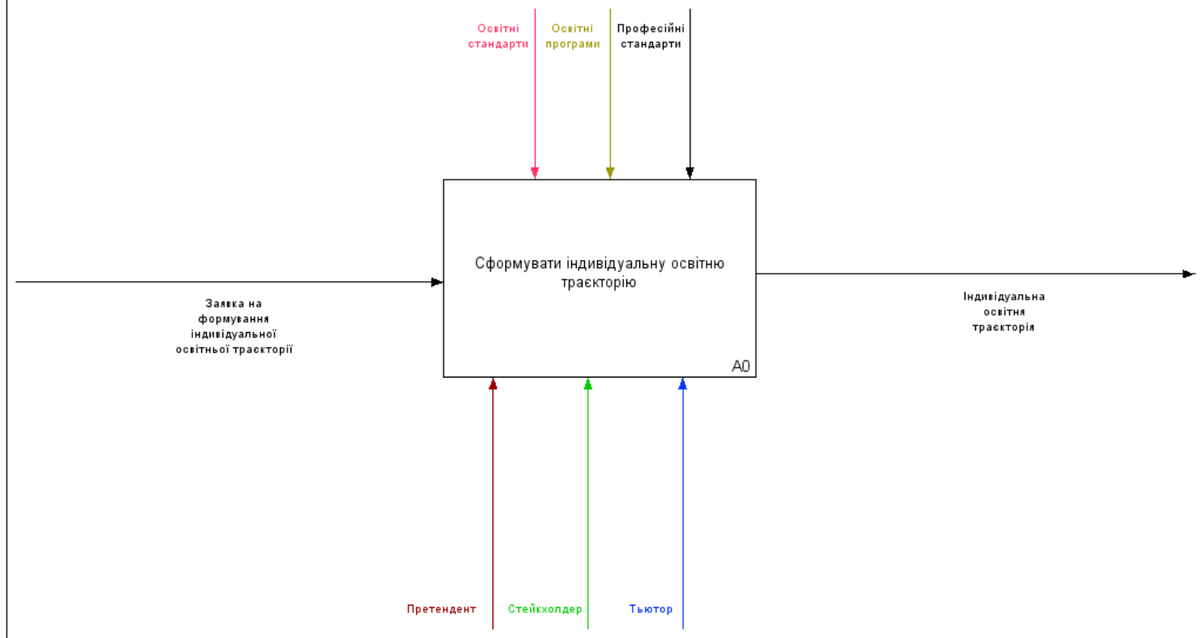
ДП ІС-5207.1181-с.СБД



					ДП ІС-5207.1181-с.СБД
Зм.	Арк.	№ документа	Підпис	Дата	Схема бази даних
Розробив		Дворник В.А.			Літера: — Маса: — Масштаб: —
Перевірив		Ковалюк Т.В.			Аркуш 1 / Аркушів 1
Т. кон.					Інформаційна система побудови індивідуальних освітніх траєкторій
Н. кон.		Халус О.А.			КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-52
Затвердив		Ковалюк Т.В.			

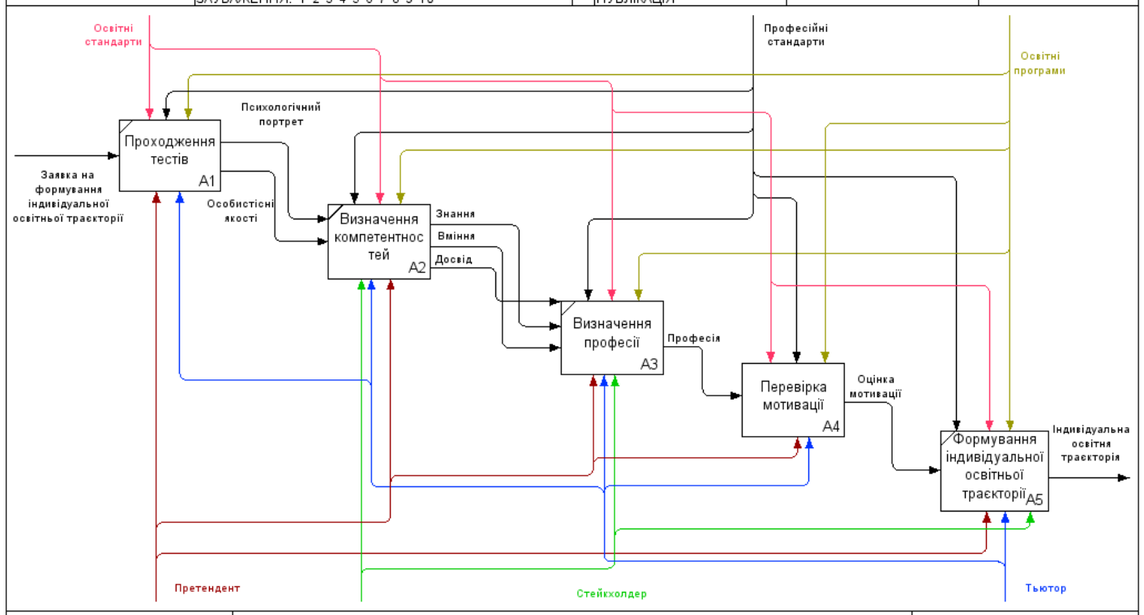
ДП ІС-5207.1181-с.ССФ

ВИКОРИСТОВУЄТЬСЯ В:	АВТОР: Дворник Вікторія	ДАТА: 24.02.2019	РОЗРОБЛЯЄТЬСЯ	ЧИТАЧ	ДАТА	КОНТЕКСТ:
	ПРОЕКТ:	РЕВІЗІЯ: 15.05.2019	ЧОРНОВИК			
			РЕКОМЕНДОВАНО			
	ЗАУВАЖЕННЯ: 1 2 3 4 5 6 7 8 9 10		ПУБЛІКАЦІЯ			



Гілка: A-0	Назва: Роботи	Номер: 1

ВИКОРИСТОВУЄТЬСЯ В:	АВТОР: Дворник Вікторія	ДАТА: 24.02.2019	РОЗРОБЛЯЄТЬСЯ	ЧИТАЧ	ДАТА	КОНТЕКСТ:
	ПРОЕКТ:	РЕВІЗІЯ: 15.05.2019	ЧОРНОВИК			
			РЕКОМЕНДОВАНО			
	ЗАУВАЖЕННЯ: 1 2 3 4 5 6 7 8 9 10		ПУБЛІКАЦІЯ			



Гілка: A0	Назва: Сформувати індивідуальну освітню траєкторію	Номер: 2

Зм.	Арк.	№ документа	Підпис	Дата	ДП ІС-5207.1181-с.ССФ		
Розробив		Дворник В.А.			Схема структурна функціональної моделі системи	Літера	Маса / Масштаб
Перевірив		Ковалюк Т.В.				Аркуш 1	Аркушів 1
Т. кон.							
Н. кон.		Халус О.А.			Інформаційна система побудови індивідуальних освітніх траєкторій	КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-52	
Затвердив		Ковалюк Т.В					

Рішення з математичного забезпечення

3.2 Математична постановка задачі

3.2.1 Задача перевірки вмотивованості студента

Нехай дана колекція з n документів, набір яких представляє собою есе, чи мотиваційний лист абітурієнта, і m різних термінів, видобутих із словника IT-термінів. Під терміном розуміється або окреме слово, або словосполучення. Для застосування моделі латентно-семантичного аналізу, потрібно побудувати $m \times n$ матрицю «термін-документ» X, з комірками $x_{i,j}$, що містять вагові коефіцієнти терміна t_i, в документі d_j. Стовпці матриці X на практиці відповідають мультимножині слів (англ. «bag-of-words») для документа, при цьому можуть бути використані терміни, «зважені» за якою-небудь схемою: наприклад, це може бути широко відома схема TF-IDF (від англ. TF – term frequency, IDF – inverse document frequency) або також можуть бути ефективні різні схеми, засновані на ентропії.

Отримана матриця «термін-документ» X являє собою просторово-векторну модель представлення текстової інформації і одночасно вхідні дані для методу латентно-семантичного аналізу [15].

Потрібно дати відповідь на питання, чи відповідає вхідний текст тематиці IT-сфери та визначити тему цього тексту, застосовуючи методи семантичного аналізу тексту.

3.2.2 Задача визначення дисциплін

Розділимо дану задачу на підзадачі:

– визначення списку областей знань для професії;

– визначення ключових слів з тематик обраних областей знань.

3.2.2.1 Підзадача визначення списку областей знань для професії

Використаємо математичну постановку задачі визначення вмотивованості студента.

Для визначення дисциплін потрібно дати відповідь на питання, як пов'язані професії та області знань, а також визначити області знань для кожної професії.

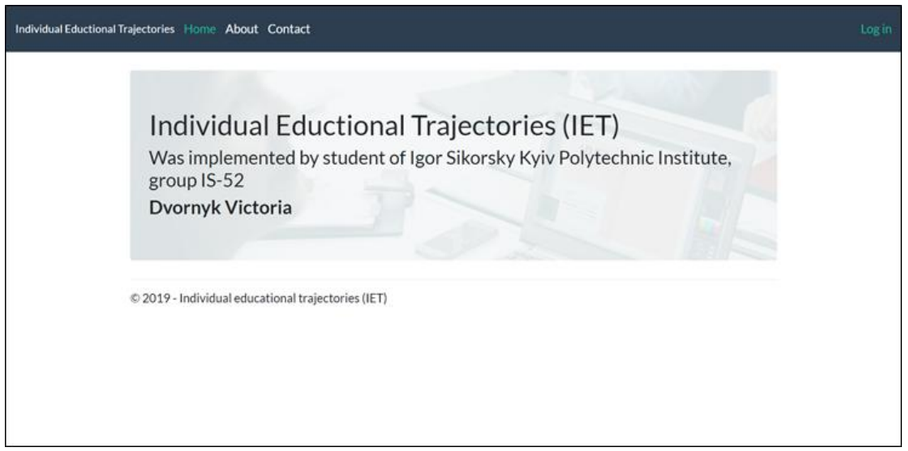
3.2.2.2 Підзадача визначення ключових слів із тематик обраних областей знань

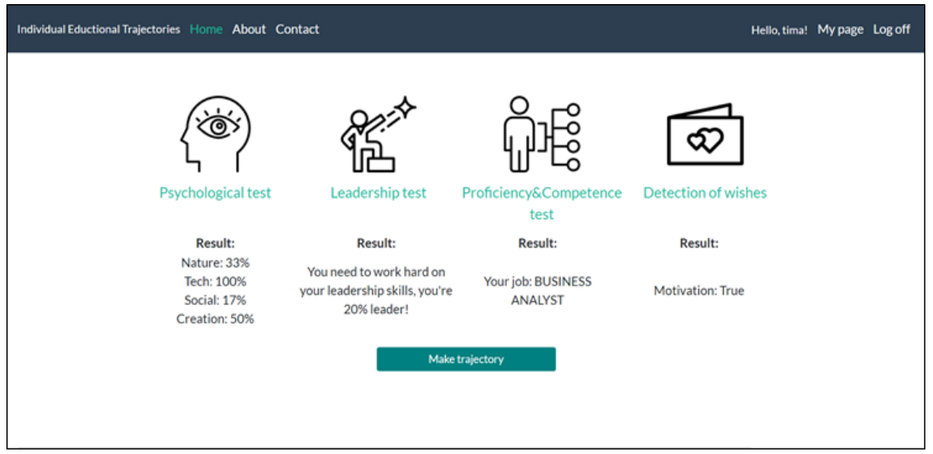
Нехай D – множина текстових документів, які являють собою тематики області знань, W – множина термінів, що вживаються в них. Кожен документ $d \in D$ представлений послідовністю термінів $\{w_i\}_{i=1}^{n_d}$ із W, де n_d – число слів у документі d. Один і той же термін може зустрічатися в документі кілька разів.

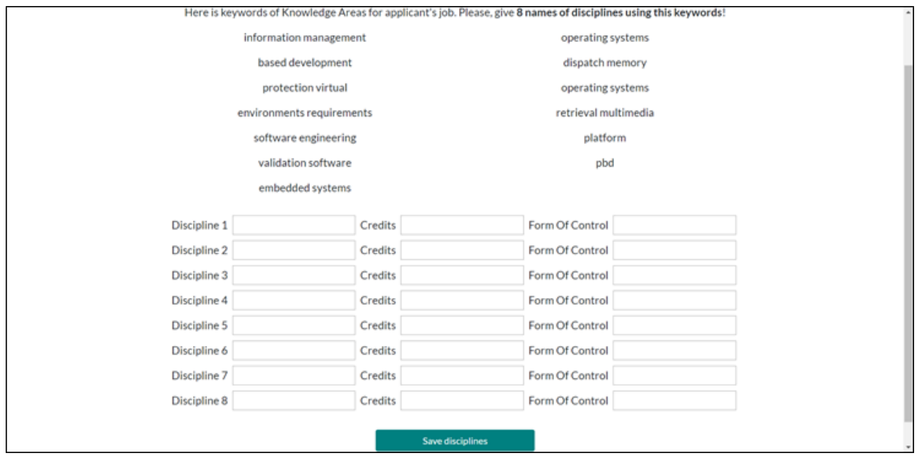
Нехай Z – це скінченна множина тематик. Нехай поява терміна w в кожному документі d пов'язана з деякою невідомою, тематикою $z \in Z$. Користуючись цим, уявімо безліч документів у вигляді безлічі трійок вигляду (d, w, z), обраних випадково і незалежно з дискретного розподілу $p(d, w, z)$, яке задано на множині $D \times W \times Z$. Незалежність елементів вибірки має на увазі те, що порядок термінів в документі не важливий для виявлення тематик.

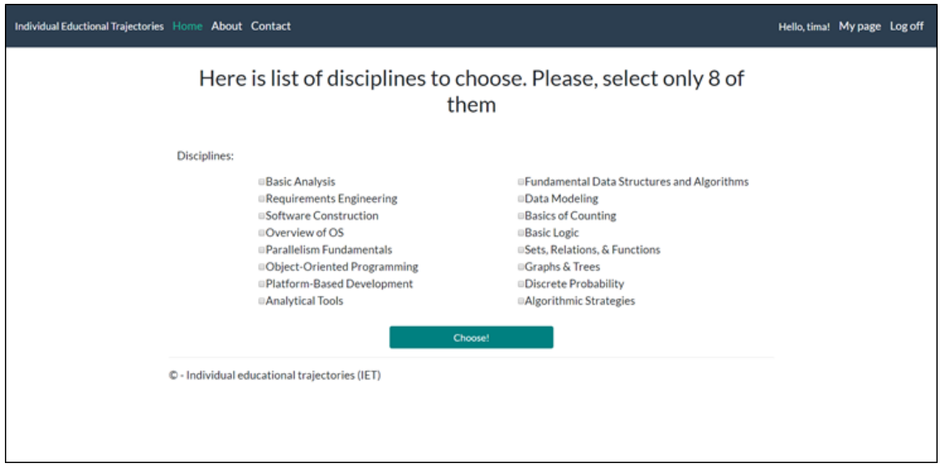
Завдання імовірнісного тематичного моделювання можна визначити наступним чином: побудувати імовірнісну тематичну модель для колекції документів D, тобто визначити множину тематик Z, ймовірність появи терму в певній темі – $p(w|z)$ для всіх тематик $z \in Z$ і ймовірність появи теми в документах – $p(z|d)$ для всіх документів $d \in D$ [16].

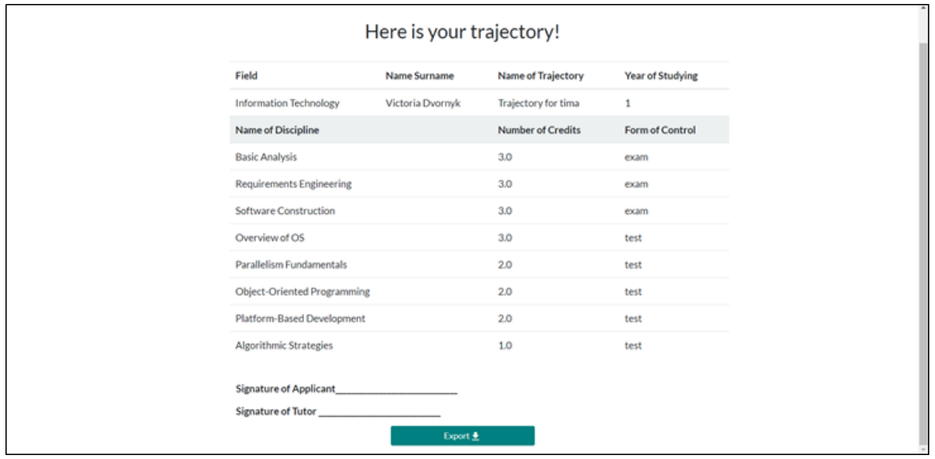
Демонстраційний плакат до дипломного проекту
„ Інформаційна система побудови індивідуальних освітніх траєкторій "
Виконала студентка гр. ІС-52 Дворник В.А.
Керівник ДП Ковалюк Т.В.











					ДП ІС-5207.1181-с.КЕ	
						Літера / Маса / Масштаб
Зм.	Арк.	№ документа	Підпис	Дата	Креслення вигляду екранних форм	
Розробив		Дворник В.А.				Аркуш 1 / Аркушів 1
Перевірив		Ковалюк Т.В.				
Т. кон.					Інформаційна система побудови індивідуальних освітніх траєкторій	КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-52
Н. кон.		Халус О.А.				
Затвердив		Ковалюк Т.В.				

					ДП ІС-5207.1181-с.ССП			
Зм.	Арк.	№ документа	Підпис	Дата	Схема структурна послідовності	Літера	Маса	Масштаб
Розробив		Дворник В.А.						
Перевірив		Ковалюк Т.В.				Аркуш 1		Аркушів 1
Т. кон.					Інформаційна система побудови індивідуальних освітніх траєкторій	КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-52		
Н. кон.		Халус О.А.						
Затвердив		Ковалюк Т.В.						